Extreme Fabric Automation Administration Guide

Version 2.4.1

9036924-01 Rev AA
March 2021

Legal Notice

Extreme Networks, Inc. reserves the right to make changes in specifications and other information contained in this document and its website without prior notice. The reader should in all cases consult representatives of Extreme Networks to determine whether any such changes have been made.

The hardware, firmware, software or any specifications described or referred to in this document are subject to change without notice.

Trademarks

Extreme Networks and the Extreme Networks logo are trademarks or registered trademarks of Extreme Networks, Inc. in the United States and/or other countries.

All other names (including any product names) mentioned in this document are the property of their respective owners and may be trademarks or registered trademarks of their respective companies/owners.

For additional information on Extreme Networks trademarks, see: www.extremenetworks.com/company/legal/trademarks

Open Source Declarations

Some software files have been licensed under certain open source or third-party licenses. End-user license agreements and open source declarations can be found at: https://www.extremenetworks.com/support/policies/open-source-declaration/

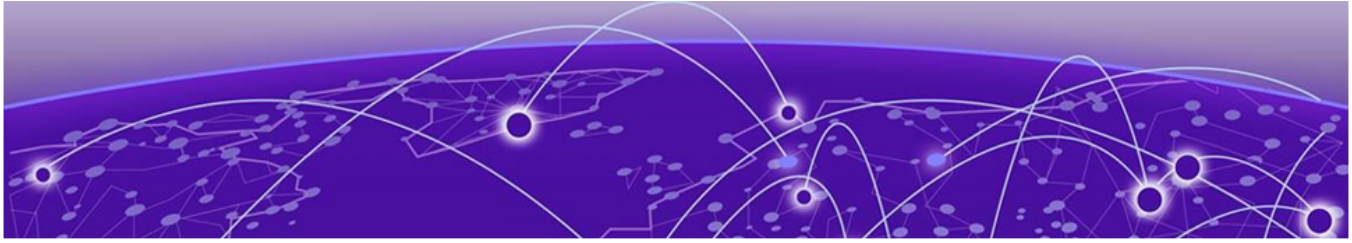


Table of Contents

Preface

This section describes the text conventions used in this document, where you can find additional information, and how you can provide feedback to us.

Text Conventions

Unless otherwise noted, information in this document applies to all supported environments for the products in question. Exceptions, like command keywords associated with a specific software version, are identified in the text.

When a feature, function, or operation pertains to a specific hardware product, the product name is used. When features, functions, and operations are the same across an entire product family, such as ExtremeSwitching switches or SLX routers, the product is referred to as *the switch* or *the router*.

Table 1: Notes and warnings

Icon	Notice type	Alerts you to...
	Tip	Helpful tips and notices for using the product
	Note	Useful information or instructions
	Important	Important features or instructions
	Caution	Risk of personal injury, system damage, or loss of data
	Warning	Risk of severe personal injury

Table 2: Text

Convention	Description
`screen displays`	This typeface indicates command syntax, or represents information as it is displayed on the screen.
The words *enter* and *type*	When you see the word *enter* in this guide, you must type something, and then press the Return or Enter key. Do not press the Return or Enter key when an instruction simply says *type*.
Key names	Key names are written in boldface, for example **Ctrl** or **Esc**. If you must press two or more keys simultaneously, the key names are linked with a plus sign (+). Example: Press **Ctrl**+**Alt**+**Del**
Words in italicized type	Italics emphasize a point or denote new terms at the place where they are defined in the text. Italics are also used when referring to publication titles.
NEW!	New information. In a PDF, this is searchable text.

Table 3: Command syntax

Convention	Description
bold text	Bold text indicates command names, keywords, and command options.
italic text	Italic text indicates variable content.
`[ ]`	Syntax components displayed within square brackets are optional. Default responses to system prompts are enclosed in square brackets.
`{ x | y | z }`	A choice of required parameters is enclosed in curly brackets separated by vertical bars. You must select one of the options.
`x | y`	A vertical bar separates mutually exclusive elements.
< >	Nonprinting characters, such as passwords, are enclosed in angle brackets.
...	Repeat the previous element, for example, *member*`[`*member*`...]`.
\	In command examples, the backslash indicates a “soft” line break. When a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash.

Documentation and Training

Find Extreme Networks product information at the following locations:

- Current Product Documentation
- Release Notes
- Hardware and software compatibility for Extreme Networks products
- Extreme Optics Compatibility
- Other resources such as white papers, data sheets, and case studies

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For details, visit www.extremenetworks.com/education/.

Getting Help

If you require assistance, contact Extreme Networks using one of the following methods:

Extreme Portal

Search the GTAC (Global Technical Assistance Center) knowledge base; manage support cases and service contracts; download software; and obtain product licensing, training, and certifications.

The Hub

A forum for Extreme Networks customers to connect with one another, answer questions, and share ideas and feedback. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.

Call GTAC

For immediate support: (800) 998 2408 (toll-free in U.S. and Canada) or 1 (408) 579 2826. For the support phone number in your country, visit: www.extremenetworks.com/support/contact

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number, or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any actions already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load at the time of trouble (if known)
- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

Subscribe to Product Announcements

You can subscribe to email notifications for product and software release announcements, Field Notices, and Vulnerability Notices.

1. Go to The Hub.
2. In the list of categories, expand the **Product Announcements** list.
3. Select a product for which you would like to receive notifications.
4. Select **Subscribe**.
5. To select additional products, return to the **Product Announcements** list and repeat steps 3 and 4.

You can modify your product selections or unsubscribe at any time.

Providing Feedback

The Information Development team at Extreme Networks has made every effort to ensure the accuracy and completeness of this document. We are always striving to improve our documentation and help you work better, so we want to hear from you. We welcome all feedback, but we especially want to know about:

- Content errors, or confusing or conflicting information.

- Improvements that would help you find relevant information in the document.
- Broken links or usability issues.

If you would like to provide feedback, you can do so in three ways:

- In a web browser, select the feedback icon and complete the online feedback form.
- Access the feedback form at https://www.extremenetworks.com/documentation-feedback/.
- Email us at documentation@extremenetworks.com.

Provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.

About this Document

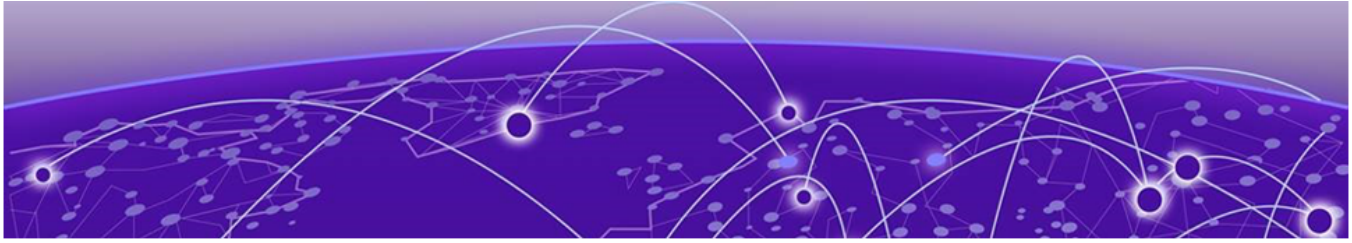
What's New in this Document on page 11

What's New in this Document

The following table describes information added to this guide for the Extreme Fabric Automation 2.4.1 software release.

Table 4: Summary of changes

Feature	Description	Link
Endpoint group configuration	You can change a BFD session formed over a CEP port from a hardware session to a software session.	Configure the BFD Session Type for an Endpoint Group on page 102



Extreme Fabric Automation

Introduction to Extreme Fabric Automation

Extreme Fabric Automation (EFA) is a micro-services-based scalable fabric automation application.

EFA automates and orchestrates SLX IP fabrics and tenant networks, with support for the following:

- Building and managing non-Clos small data center fabrics and 3-stage and 5-stage IP Clos fabrics
- Managing tenant-aware Layer 2 and Layer 3 networks
- Configuring integration with several ecosystems: VMware vCenter, OpenStack, and Microsoft Hyper-V
- Providing a single point of configuration for your entire fabric

EFA consists of core K3s containerized services that interact with each other and with other infrastructure services to provide the core functions of fabric and tenant network automation. For more information, see EFA Microservices on page 14.

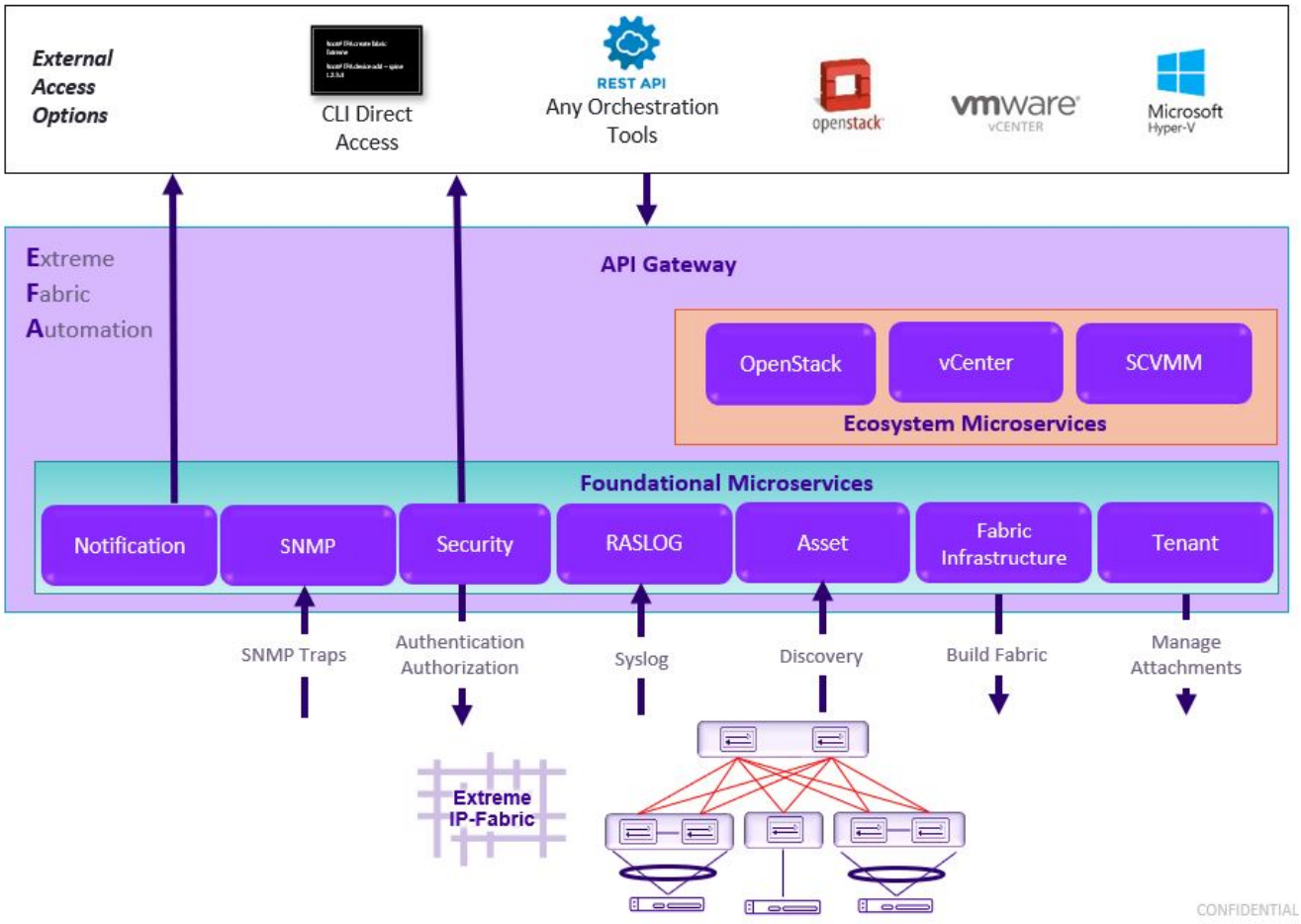


Figure 1: EFA orchestration

CLI and API

Using the built-in command and OpenAPI-based REST APIs, you can discover physical and logical assets, build and manage fabrics, manage the EFA system, and configure security. For more information, see the Extreme Fabric Automation Command Reference, 2.4.0 and REST API Documentation for EFA on page 17.

Deployment

For more information about deployment scenarios, see the Extreme Fabric Automation Deployment Guide, 2.4.0.

EFA on TPVM

TPVM (Third-Party Virtual Machine) is a guest VM that resides on Extreme SLX devices. You can run EFA from the SLX 9150, SLX 9250, or SLX 9740 TPVM. In this context, EFA leverages the K3S Kubernetes cluster as an underlying infrastructure for the EFA services deployment. The K3S cluster is a single instance and an important component for supporting high availability. A maximum of 24 devices is supported, either 24 devices in one fabric or 24 devices across multiple fabrics.

EFA on an external VM

You can deploy EFA on an external Virtual Machine to support more than 24 devices or based on where tools are deployed in the data center.

EFA for high availability

A high-availability cluster is a group of servers that provide continuous up time, or at least minimum down time, for the applications on the servers in the group. If an application on one server fails, another server in the cluster maintains the availability of the service or application. You can install EFA on a two-node cluster, including on TPVM, for high availability.

EFA Microservices

EFA consists of core K3s containerized microservices that interact with each other and with other infrastructure services to provide the core functions of fabric and tenant network automation.

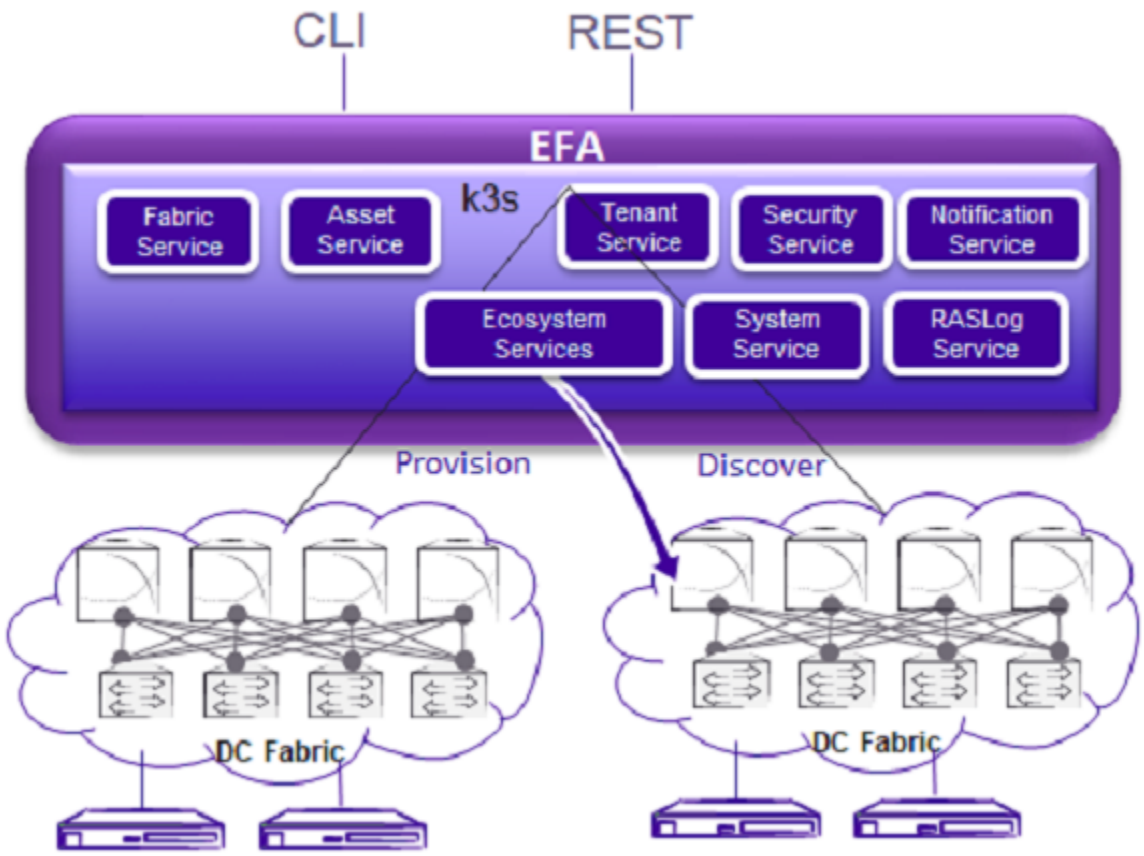


Figure 2: Microservices in the EFA architecture

Fabric Service

The Fabric Service is responsible for automating the fabric BGP underlay and EVPN overlay. By default, the EVPN overlay is enabled but you can disable it before provisioning, if necessary. The Fabric Service exposes the CLI and REST API for automating the fabric underlay and overlay configuration.

The Fabric Service features include:

- Support for small data centers (non-Clos)
- Support for 3-stage and 5-stage Clos fabrics
- Support for MCT configuration

Underlay automation includes interface configurations (IP numbered), BGP underlay for spine and leaf, BFD, and MCT configurations. Overlay automation includes EVPN and overlay gateway configuration.

Tenant Service

The Tenant Service manages tenants, tenant networks, and endpoints, fully leveraging the knowledge of assets and the underlying fabric. You can use the CLI and REST API for tenant network configuration on Clos and non-Clos fabrics.

Tenant network configuration includes VLAN, BD, VE, EVPN, VTEP, VRF, and router BGP configuration on fabric devices to provide Layer 2 extension, Layer 3 extension across the fabric, Layer 2 hand-off, and Layer 3 hand-off at the edge of the fabric.

Inventory Service

The Inventory Service acts as an inventory of all the necessary physical and logical assets of the fabric devices. All other EFA services rely on asset data for their configuration automation. The Inventory Service is a REST layer on top of device inventory details, with the capability to filter data based on certain fields. The Inventory Service securely stores the credentials of devices in encrypted form and makes those credentials available to different components such as the Fabric and Tenant services.

The Inventory Service supports the **`execute-cli`** option for pushing configuration and exec commands to devices. Examples include configuring SNMP parameters or OSPF configurations. This means you can use EFA for SLX-OS commands and push the same configuration to multiple devices.

The Asset Service provides the secure credential store and deep discovery of physical and logical assets of the managed devices. The service publishes the Asset refresh and change events to other services.

Notification Service

The Notification Service sends events, alerts, and tasks to external entities. Notifications sent from EFA are derived from the syslog events received from the devices that EFA manages. Alerts are notifications that services in EFA send for unexpected conditions. Tasks are user-driven operations or timer-based tasks such as device registration or fabric creation.

RASlog Service

The RASlog Service processes syslog messages from devices and forwards notifications to subscribers. For more information, see RASlog Service in the Extreme Fabric Automation Administration Guide, 2.4.0.

Security Service

The Security Service consists of authentication and authorization features that enforce a security boundary between northbound clients and downstream operations between EFA and SLX devices. The service also validates users and their credentials through Role-based Access Control (RBAC) and supports local and remote (LDAP) login.

SNMP Service

The SNMP Service processes SNMP traps from devices and forwards notifications to subscribers. For more information, see EFA as SNMP Proxy in the Extreme Fabric Automation Administration Guide, 2.4.0..

Ecosystem Integration Services

EFA provides one-touch integration with these ecosystems, providing deep insight into VMs, vSwitches, port groups, and hosts, and the translation of these into IP fabric networking constructs.

VMware vCenter Service

The vCenter integration provides connectivity between EFA and vCenter using a REST API. EFA does not connect to individual ESXi servers. All integration is done through vCenter. For more information, see the Extreme Fabric Automation vCenter Integration Guide, 2.4.0. Integration support includes the following:

- Registration or deregistration of one or more vCenter servers in EFA
- Updates for vCenter asset details
- Lists of information about vCenter servers
- Inventory integration
- Dynamic updates about Tenant Service integration from vCenter and from EFA services

Hyper-V

The Hyper-V integration supports networking configuration for Hyper-V servers in a datacenter, manual and automated configuration updates when VMs move, and visibility into the VMs and networking resources that are deployed in the Hyper-V setup. For more information, see Extreme Fabric Automation Hyper-V Integration Guide, 2.4.0. Integration support includes the following:

- SCVMM (System Center Virtual Machine Manager) server discovery
- SCVMM server update
- Periodic polling of registered SCVMM servers
- SCVMM server list
- SCVMM server delete and deregister
- Network event handling

OpenStack Service

The OpenStack service integrates Extreme OpenStack plug-ins with the rest of the EFA foundation services in an IP fabric. For more information, see the Extreme Fabric Automation OpenStack Integration Guide, 2.4.0. Integration support includes the following:

- Create, read, update, delete (CRUD) operations on networks and ports
- LAG support

- Provider network (default, PT)
- VLAN trunking
- Network operations using single-root I/O virtualization (SR-IOV), physical and virtual functions
- vMotion (virtual machine migration)
- ML2 driver with support for:
  - Network and segment provisoning for non-default physnets
  - DC-owner-based l2 extension for DC gateway
- Topology changes for port-based extension of DC gateway addition and deletion of topology entries and its changes on EFA EPGs
- Single-homed connections to the edge port
- Multi-segment support
- Journaling support for L2 and L3
- L3 service plugin:
  - Routing feature support using VRF
  - Flavor (service provider) support
  - Centralized routing
  - IPv6 support (dual stack)
- Layer 3 flavors
- Neighbor Discovery and Router Advertisement support:
  - IPv6 ND MTU support
  - IPv6 No-Autoconfig support

REST API Documentation for EFA

When EFA is installed, REST API documentation is available as an HTML reference: `http://<host_ip>/docs`.

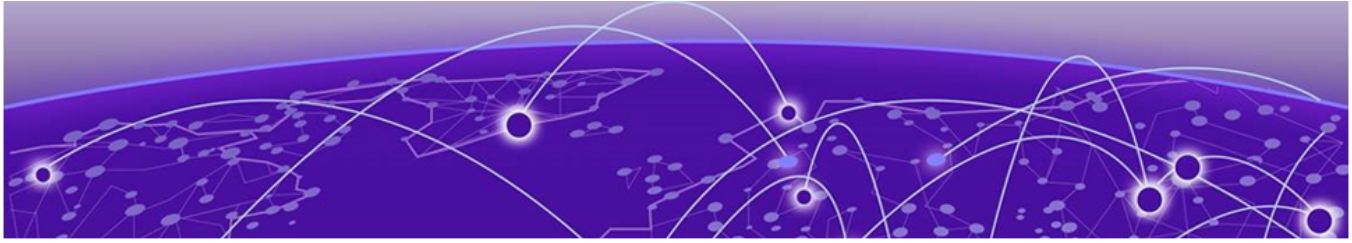
The REST API is specified by OpenAPI and Swagger.

Specific API guides for the EFA services are available on the Extreme Networks website. Select **Extreme Fabric Automation** here: https://www.extremenetworks.com/support/documentation/product-type/software/. And then select the version of EFA you want to work with.

API guides are available for the following services:

- Authorization service
- Fabric service
- Hyper-V service
- Inventory service
- Monitoring service
- Notification service
- OpenStack service
- RASlog service
- RBAC service
- SNMP service

- System service
- Tenant service
- vCenter service



EFA System Management

Verify the Running System and Services

After any of the following scenarios, wait 10 minutes for EFA micro-services to be operational before you run EFA commands.

- Powering on the OVA
- Rebooting the OVA
- Rebooting the TPVM
- Rebooting the SLX (which also reboots the TPVM)
- Rebooting the server on which the EFA is installed

You can use various commands and scripts to verify the status of the EFA system, to help troubleshoot, and to view details of EFA nodes, PODs, and services.

1. Verify the K3s installation in a TPVM.
   a. Run the **`show efa status`** command from the SLX command prompt.

```
device# show efa status
       NAME    STATUS    ROLES     AGE      VERSION
    TPVM    Ready     primary    6m59s    v1.14.5-k3s.1
    admin@10.24.51.226's password:
    NAME                              READY       STATUS     RESTARTS    AGE
    pod/godb-service-wk57h            1/1          Running   0           6m11s
    pod/gofabric-service-8v8b2        1/1          Running   3           6m12s
    pod/goinventory-service-4kggf     1/1          Running   3           6m12s
```

```
    pod/gotenant-service-xcqf6      1/1         Running  3          6m12s
    pod/rabbitmq-0                  1/1         Running  0          6m12s
    pod/rabbitmq-1                  1/1         Running   0         4m51s
```

Output varies by type of deployment, such as single-node or multi-node, and the services that are installed.

2. View details of EFA nodes, PODs, and services.
   a. Run the **efa status** command.

      On a multi-node installation:

```
$ efa status
+-----------+---------+--------+
| Node Name | Role    | Status |
+-----------+---------+--------+
| tpvm      | active  | up     |
+-----------+---------+--------+
| tpvm2     | standby | up     |
+-----------+---------+--------+
--- Time Elapsed: 2.318350499s ---
```

      On a single-node installation:

```
$ efa status
+-----------+---------+--------+
| Node Name | Role    | Status |
+-----------+---------+--------+
| tpvm      | active  | up     |
+-----------+---------+--------+
--- Time Elapsed: 2.224282775s ---
```

      These examples show only a few of all possible rows of detail.

3. Verify that all PODs are in a running state.
   a. Run the **k3s kubectl get pods -n efa** command.

```
# k3s kubectl get pods -n efa

NAME                                    READY   STATUS    RESTARTS   AGE
goswitch-service-958fcfb4f-qddnw        1/1     Running   4          72d
godb-service-57bd99747-f4cxb            1/1     Running   4          83d
efa-api-docs-6bb5dbcc74-br485           1/1     Running   4          72d
filebeat-service-86ddd654b6-z9zhr       1/1     Running   4          72d
goopenstack-service-554c57548f-bjwtb    1/1     Running   8          72d
rabbitmq-0                              1/1     Running   7          72d
govcenter-service-f6b49d9b9-s24wk       1/1     Running   19         72d
gohyperv-service-854654f6b9-m9mv8       1/1     Running   20         72d
goinventory-service-59d9b798d8-s9wn6    1/1     Running   20         72d
gotenant-service-55fd8889d8-g8rgb       1/1     Running   19         72d
gofabric-service-69d8995fc6-swnqw       1/1     Running   19         72d
metricbeat-service-76c4874887-mbm7h     1/1     Running   32         72d
```

      In a multi-node installation, only the pods on the active node are in "Running" status.

4. Verify the status of the Authentication service.
   a. Run the **systemctl status hostauth.service** script.

```
$ systemctl status hostauth.service
hostauth.service - OS Auth Service
Loaded: loaded (/lib/systemd/system/hostauth.service; enabled; vendor preset:
enabled)
Active: active (running) since Thu 2020-04-23 07:56:20 UTC; 23 h ago
Main PID: 23839 (hostauth)
Tasks: 5
CGroup: /system.slice/hostauth.service
```

```
         23839 /apps/bin/hostauth

Apr 23 07:56:20 tpvm2 systemd[1]: Started OS Auth Service
```

5. Restart a service using the **`efactl restart-service <service-name>`** command.
6. Identify the active node that serves as the database for Kubenetes clusters.
   a. Run the **`ip addr show`** command from all nodes.
   b. Verify that on one of the Ethernet interfaces, the virtual IP address shows up as the secondary IP address.

Log in to EFA

Use of the EFA command line requires a valid, logged-in user.

1. Verify the status of the EFA deployment using one of the following methods.
   - Run the SLX **`show efa status`** command.
   - Run the EFA **`efactl status`** script (or the **`efa status`** command, as an alternative).

   For more information, see Verify the Running System and Services on page 19.
2. Log in to EFA.

   ```
   $ efa login --username <username>
   Password: <password>
   ```

   The <username> variable is optional. If you do not provide a user name, log-in defaults to the current (Unix) user.

   With a successful log-in, the command prompt shows the logged-in user in green text. If the log-in is not successful, the command prompt is displayed in red text.
3. To log out of EFA, run the **`efa logout`** command.

EFA User Authentication and Authorization

EFA users are validated with Unix authentication or LDAP and managed with Role-based Access Control (RBAC).

For more information, see Assign and View EFA Roles on page 26 and EFA RBAC Policy Enforcement on page 23.

Authentication

EFA validates users and their credentials with the following mechanisms:

- Unix authentication (local and remote) on the host where EFA is installed. Host credentials are the default validation method if LDAP validation fails.
- External LDAP server. Users configured in LDAP use their LDAP credentials to log in to EFA.

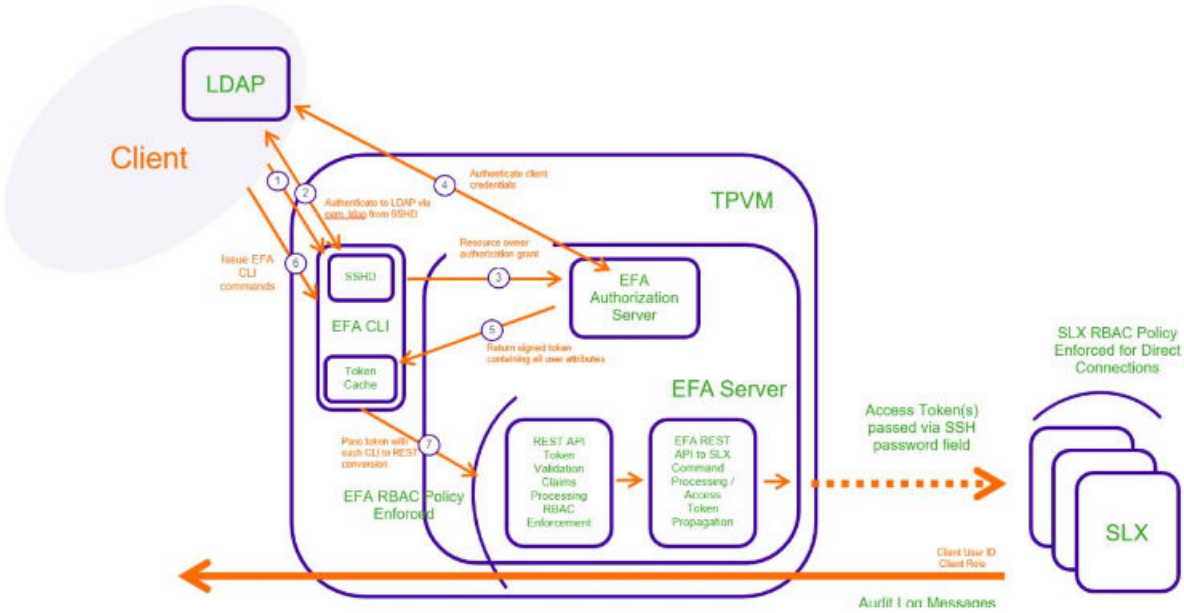


Figure 3: LDAP authentication example

Users who perform operational or maintenance tasks are propagated to SLX devices through OAuth2 and JWT access tokens. TLS is used for connections with SLX devices. The OpenStack ML2 plugin also uses TLS and OAuth2 tokens. When EFA is installed in secure mode, traffic to northbound interfaces uses TLS. For more information about secure mode, see the "EFA Installation Modes" topic in the Extreme Fabric Automation Deployment Guide, 2.4.0.

Authorization

After EFA is deployed, the installing user has the role of SystemAdmin and has complete access to EFA functionality. For installation on TPVM, this user has the user name of ‘extreme’. By default, no other host OS users can access EFA unless the SystemAdmin assigns the appropriate roles. RBAC occurs on EFA and API.

LDAP supports three modes for fetching the roles assigned to a user.

- The role is available as an attribute in the user Distinguished Name (DN) entry. Group attribute definition is not needed.
- The user has a "memberOf" attribute or any appropriate group DN attribute to identify the groups assigned to the user. Assign the corresponding LDAP group to a role in EFA.
- LDAP groups have user entries in their group definitions. Assign the LDAP groups to roles in EFA.

Security troubleshooting

Use the following logs to troubleshoot authentication, authorization, or RBAC issues.

Table 5: Security log locations

Log source	Filepath
EFA server	`/var/log/efa/auth/auth-server.log` `/var/log/efa/rbac/rbac-server.log`
EFA TPVM	`/apps/efa_logs/auth/auth-server.log` `/apps/efa_logs/rbac/rbac-server.log`
SLX device	`/var/log/pam-oauth2.log`

Use the following commands to see lists of commands that were run during a specified time, such as when an RBAC error occurred. This sort of information can help you identify potential causes.

- **`efa auth execution show`**
- **`efa rbac execution show`**
- **`efa inventory execution show`**

EFA RBAC Policy Enforcement

EFA implements an RBAC (Role-based Access Control) policy governing access to northbound REST APIs.

The RBAC policy is enforced at the northbound interface, immediately after validation of the access token. An error message is returned if an RBAC permissions check fails.

RBAC and REST URI matrix

The RBAC policy is expressed in a permissions matrix indexed by RBAC role and REST URI, in which each matrix element enumerates the permitted HTTP methods.

Table 6: RBAC and REST matrix

	Role A	Role B	Role C
REST URI 1	GET	GET	GET, POST, PUT, PATCH, DELETE
REST URI 2	GET, POST	GET, POST, PUT	GET, POST, PUT, PATCH, DELETE
REST URI 3	GET, POST	GET, POST	GET, POST, PUT, PATCH, DELETE

RBAC roles

Roles can be populated into the upstream LDAP instance.

Table 7: Role definitions

Role	Description
FabricAdmin	• Registers devices to the fabric • Configures fabric parameters • Validates all devices in the fabric • Configures switches for IP fabric with overlay and without overlay • Creates tenants • Creates networks inside tenants, such as VRF, EPG, and PO • Performs fabric debug activities • Has privileges for OpenStack, Hyper-V, and vCenter operations
SecurityAdmin	Performs user management, PKI, and key management operations
NetworkOperator	• Has view-only privileges for fabric configurations, information for tenants and inventory, and all ecosystem information • Cannot make changes in the system
SystemDebugger	• Has privileges to perform supportsave and system backup, and to view the running system configurations • Has privileges to perform fabric debug operations • Sets debug levels for services • Has privileges to collect execution logs from services
SystemAdmin	Has complete privileges to all operations in the system
<Tenant>Admin * Created dynamically per tenant	Performs tenant administration within the assigned tenant, such as the following: • Adding networks to the tenant • Configuring network parameters • Configuring switches with tenant-specific information Cannot perform actions for any other tenant

* Tenant Administrator roles are added dynamically to the system when a tenant is created. The name of the role is of the format `<Tenant-name>Admin`. For example, if a tenant with the name "RegionOne" is created, the role created for the Tenant Administrator is "RegionOneAdmin".

Note
You cannot create custom roles.

Role permissions

Allowed Privileges	System Admin	Fabric Admin	Tenant Admin	Network Operator	Security Admin	System Debugger
Create/Clone/Delete fabric in the system	✔	✔				
Register/Unregister devices in fabric, configure IP fabric on the device	✔	✔				
Show IP fabric physical/underlay/ overlay topology, IP fabric configs and devices in IP fabric	✔	✔		✔		
Debug fabric operations	✔	✔				✔
Inventory/Asset service operations	✔	✔				
Run CLI access on the device	✔	✔				
Create/Delete/Update tenants	✔	✔				
Create/Delete EPG, PO, VRFs inside tenant	✔	✔	✔			
Add/Remove Port/Port Channels to/from EPG	✔	✔	✔			
Add/Remove Network Policies to EPG	✔	✔	✔			
Detach Network from EPG	✔	✔	✔			
Identify drift in device configuration	✔	✔				
Set tenant debug level	✔	✔	✔			✔
Show OpenStack networks, PO, subnets, tenant, ports, router, router-interface	✔	✔	✔	✔		
Create/Delete/Cleanup OpenStack Networks	✔	✔	✔			
Create/Delete OpenStack Subnets	✔	✔	✔			
Create/Delete OpenStack Ports	✔	✔	✔			
Create/Delete OpenStack Router	✔	✔	✔			
Create/Delete Router Interfaces	✔	✔	✔			
Delete OpenStack asset (DebugDeleteOSSAsset)	✔	✔	✔			✔
View vCenter Details, events, ESXI details, Physical links, Virtual Links, Disconnected links, Get server settings	✔	✔	✔	✔		
Register/Delete/Update vCenter	✔	✔	✔			
Set vCenter debug level	✔	✔	✔			✔

Update vCenter polling frequency, dead link clearing time	✔	✔	✔			
View SCVMM server details, Service Settings, Physical Links, Virtual Links	✔	✔	✔	✔		
Register/Delete/Update SCVMM server	✔	✔	✔			
Update SCVMM server polling frequency	✔	✔	✔			
User Management, assign roles to users, configure LDAP, view available roles in the system	✔				✔	
Notification service (Add/Delete subscribers)	✔	✔				
Execution Log View	✔	✔	✔ (Only Tenant)	✔	✔ (Only Auth and RBAC)	✔
Support Save Collection	✔	✔	✔	✔	✔	✔
Backup and Restore Operation	✔	✔				✔ (Only backup)
Install certificates	✔	✔			✔	

Assign and View EFA Roles

You can assign a role to a user and to an LDAP group.

For more information about EFA roles, see EFA RBAC Policy Enforcement on page 23.

1. To assign a role to a user, run the following command.

```
# efa auth rolemapping add --name fabricuser --role FabricAdmin --type user
Successfully added the role mapping
```

   In this example, a user named fabricuser was assigned the role of FabricAdmin.

2. To assign a role to an LDAP group, run the following command.

```
# efa auth rolemapping add --name "cn=viewer,dc=extr,dc=com" --role NetworkOperator
--type group
Successfully added the role mapping.
```

   In this example, a group named "cn=viewer,dc=extr,dc=com" was assigned the role of NetworkOperator.

3. To view all role assignments, run the following command.

```
# efa auth rolemapping show
ID  Name     Role          Type
1   efauser    SystemAdmin      USER
2   fabricuser FabricAdmin      USER
3   viewer     NetworkOperator  GROUP
```

4. To delete a role assignment, run the following command

```
# efa auth rolemapping remove --id 3
Deleted role mapping successfully
```

In this example, the role for the user with ID 3 was removed.

Configure an External LDAP Server

You configure an LDAP server for user validation and to fetch user groups.

For more information about commands and supported parameters, see Extreme Fabric Automation Command Reference Version 2.4.0.

1. To configure an external LDAP server, run the following command.

```
# efa auth ldapconfig add --name ldapconfig -- host 10.x.x.x --bind-user-
name cn=admin,dc=extrnet,dc=com --bind-user-password password --user-search-
base ou=people,dc=extrnet,dc=com
```

This example configures the bind user name and password and the DN of the node from which searches start.

2. To configure an LDAP server in a TPVM (for the TPVM Ubuntu OS), run the **`tpvm config ldap`** command from the SLX-OS command line.

EFA Certificate Management

EFA requires certificates for the northbound interface and certificates for devices.

Northbound interface certificates

The certificate is bundled with EFA and signed by the private Certificate Authority (CA) Chain. So that the certificate can be replaced with a third-party certificate acquired through trusted CAs (such as Verisign or GoDaddy), the certificate must be present in the host device that is running EFA. You can then install it with the following command:

```
$ efa certificates server --certificate <cert-filename>
--key <key-filename> [ --configfile  <config-filename ]
```

The `EFA_INSTALL_DIR` environment variable specifies where the EFA configuration file can be found. The optional configuration file can be used to specify a different file than the `efa.conf` file used by EFA for its settings.

Important
If you install your own server certificate to use with the EFA HTTPS server, remember to reinstall the certificate when you upgrade EFA.

Communication with third-party certificates in an EFA installation is enabled on the following ports:

- 443: Secure installation of EFA
- 8078: Monitoring service of EFA

For information about third-party certificates in a multiple management IP network, see Configuration Supporting Multiple Management IP Networks on page 48.

For a multi-node deployment, EFA uses the common name (CN) of the virtual IP address and a Subject Alternate Name containing the virtual IP address and the node IP addresses.

Example for a single-node deployment:

```
Subject: CN=efa.extremenetworks.com
          ……
             X509v3 Subject Alternative Name:
                 DNS:efa.extremenetworks.com, IP Address:127.0.0.1,
IP Address:10.24.15.173
```

Example for a multi-node deployment:

```
  Subject: CN=efa.extremenetworks.com
         ……
             X509v3 Subject Alternative Name:
                 DNS:efa.extremenetworks.com, IP Address:127.0.0.1, IP
Address:10.24.15.178,
IP Address:10.24.15.174, IP Address:10.24.15.253
```

Device certificates

The HTTPS server certificate from EFA is presented to a client when that client connects to its northbound interface.

During the registration of an SLX device in EFA, the following configuration changes are made on the device.

- The public certificate for verifying an EFA token is copied to the device as an OAuth2 certificate.
- A syslog certificate is installed on the device.
- EFA generates the HTTPS certificate for the SLX device. The certificate is copied to the device, HTTP mode is disabled on the device, and HTTPS is enabled on the device.
- OAuth2 is enabled as the primary mode of authentication. Fallback is set to "local login."
- Managed devices contain the expiration date of an inventory certificate. If a device certificate is within 30 days of expiration, it pushes an event to the notification using the **`get certs`** command.

You can use the **`efa inventory device list`** command to verify the status of the certificates on the device. If the Cert/Key Saved column contains "N," then certificates are not installed.

You can use the **`efa certificates device install --ips <ip-adddr> certType [ http|token]`** command to install the HTTPS or OAuth2 certificate on one or more devices.

Certificate troubleshooting

Issue	Resolution
My device is registered but the certificates do not appear on the SLX device.	Try the following: • Ensure that the device is running at least SLX-OS 20.1.x. • Ensure that the time on the SLX device and the time on the EFA host device are synchronized. • Ensure that the certificates are installed. Run the **`efa certificates device install`** command.
How do I verify the certificate provided by EFA through its ingress interface?	Run the following command. The output should indicate that `efa.extremenetworks.com` is present. `$ openssl s_client -connect <EFA_IP_ADDR>:443`

Monitoring EFA Status

The Monitoring service provides REST API to monitor the status of the various services running in EFA.

The service runs on the host and is exposed on port 8078, which is not the port where the EFA application is running. In a multi-node deployment, this service is available on both nodes and can be accessed through the virtual IP (VIP).

- To start or stop the Monitoring service, run the **`systemctl stop/start/restart monitor.service`** command as a sudo or root user.

  For information, see REST API Documentation for EFA on page 17.
- Use the **`efa status`** command to verify application status.

  For more information, see the **`efa status`** command in the Extreme Fabric Automation Command Reference, 2.4.0.

Verifying EFA System Health

This topic describes methods for verifying the health of various EFA services.

SLX device health

By default, health check functionality is deactivated when SLX devices are registered. You can verify the status of the functionality with the following EFA command.

```
$ efa inventory device setting show --ip <ip-addr>

+---------------------------------+-------+
| NAME | VALUE |
+---------------------------------+-------+
| Maintenance Mode Enable On | No |
| Reboot | |
+---------------------------------+-------+
| Maintenance Mode Enable | No |
+---------------------------------+-------+
```

```
| Health Check Enabled | No |
+--------------------------------+-------+
| Health Check Interval | 6m |
+--------------------------------+-------+
| Health Check Heartbeat Miss | 2 |
| Threshold | |
+--------------------------------+-------+
| Periodic Backup Enabled | Yes |
+--------------------------------+-------+
| Config Backup Interval | 24h |
+--------------------------------+-------+
| Config Backup Count | 4 |
+--------------------------------+-------+
--- Time Elapsed: 270.251797ms ---
```

You can enable health check functionality on the device. And you can configure EFA to regularly back up the device configuration (every 6 minutes by default). For more information, see Configure Backup and Replay on page 161.

If the threshold for missed heartbeats is exceeded, EFA begins the drift and reconcile process after connectivity to the device is re-established. For more information, see Drift and Reconcile on page 39.

EFA services health

All services in EFA have internal health REST APIs that Kubernetes uses to restart pods that are deemed unhealthy. The results of a liveness probe determines whether a pod is healthy. Typical values for liveness probes are as follows:

- initialDelaySeconds: 60
- periodSeconds: 10
- timeoutSeconds: 15

RabbitMQ liveness

The EFA message bus is the workhorse for asynchronous inter-service communication. Therefore, EFA uses the RabbitMQ built-in ping functionality to determine the liveness of the RabbitMQ pod.

As part of a health check, each EFA service also validates its connection to RabbitMQ and attempts to reconnect to RabbitMQ when necessary.

EFA system health for high-availability deployments

During of installation or upgrade of EFA, a systemd service called `efamonitor` is set up. This service validates every minute during which the EFA database cluster, Kubernetes cluster, and RabbitMQ cluster are formed and functioning correctly.

As needed, the `efamonitor` service remediates the MariaDB Galera and K3s clusters, and deletes pods that are not in the correct deployment state. Finally, the service reforms the RabbitMQ cluster if a split brain state occurs.

Node health

To ensure that the active and standby nodes are operational, ping checks occur between the nodes. The pings determine whether the active node is up and running. If not, the virtual IP addresses are switched over to the other node.

To ensure that failover does not occur due to a network issue, if a ping to the peer fails, a ping is also attempted to the default gateway. If ping to default gateway fails, ping is attempted to any alternative gateway that may have been provided during installation or upgrade.

If all of the pings fail, keepalived triggers Kubernetes to switch over to the active node and to put the other node in a Fault state.

EFA System Backup and Restoration

The backup process saves EFA data, including the database, certificates, and multi-access network configuration. The process does not back up northbound certificates.

Manual backup and restore

The backup process creates a backup tar file. You can select from all saved tar files during the restore process. The tar files are saved to one of the following locations.

- Server: `/var/log/efa/backup`
- TPVM: `/apps/efa_logs/backup`

A backup generated on one EFA system can be restored on another system.

For more information, see Back up and Restore the EFA System on page 32.

You can use the **`efa system backup-list`** command to see the backup files that are available to use in a restore operation. For example:

```
$ efa system backup-list
+----+-----------------------------------+---------+----------------+
| ID    | File                           | Version | Generated By   |
+----+-----------------------------------+---------+----------------+
| 1     | EFA-2020.08.20-20.26.46.tar    | 2.3.0-1 | User           |
+----+-----------------------------------+---------+----------------+
| 2     | EFA-2020.08.20-20.27.29.tar    | 2.3.0-GA| System         |
+----+-----------------------------------+---------+----------------+
--- Time Elapsed: 183.69386ms ---
```

Periodic backups and configuration

EFA periodically backs up the system. The periodic backup process creates a backup tar file that is saved to the same location as the manual backup files. When new backup and supportsave files are created, the system deletes saved system-generated backup files, supportsave files, and manual backup files, according to the age of the files and the configured maximum number of files to save.

You can use the **`efa system settings update`** command to determine the backup schedule and to change the maximum number of backup files to save. The default is 5 backup files and 5 supportsave files. For more information, see the Extreme Fabric Automation Command Reference, 2.4.0.

You can use the **efa system settings show** command to view the current backup settings.

Tip
You can use the **efa system cleanup** REST API to delete a specified backup or supportsave file. This feature lets you delete files before they are automatically deleted.

Backups during upgrades

The process of upgrading EFA also backs up the EFA system, so that you can easily recover data if the upgrade fails. For more information, see "Recover from an Upgrade Failure" in the Extreme Fabric Automation Deployment Guide, 2.4.0.

Logs

Logs related to backup, restore, and supportsave operations are saved to the following locations.

- Server: `/var/log/efa/monitor`
- TPVM: `/apps/efa_logs/system`

The REST APIs for backup, restore, and supportsave are part of the Monitoring service and can be accessed through port 8078. The logs for these APIs are saved to `<log_dir>/monitor/`

Back up and Restore the EFA System

You can back up and restore the EFA system, including the database and certificates.

1. To back up the system, run the following command.

```
$ efa system backup
Generating backup of EFA...
Backup Location: /apps/efa_logs/backup/EFA-2021-02-10T13-21-46.413.tar
--- Time Elapsed: 10.401999384s ---
```

2. To restore the system, take the following steps.
   a. Run the **efa system restore command**.

```
$ efa system restore

EFA-2021.01.12-11.19.52.tar (Version:2.3.2-GA, Generated by: User)
EFA-2021-01-12T04.59.09.tar (Version:2.4.0-7171, Generated by: User)
EFA-2021-01-12T13.50.00.tar (Version:2.4.0-121, Generated by: User)
EFA-2021-01-12T13.50.51.tar (Version:2.4.0-121, Generated by: System)
EFA-2021-01-12T16.11.47.tar (Version:2.4.0-1211, Generated by: System)
EFA-Upgrade-2.3.2-GA.tar (Version:2.3.2-GA, Generated by: System)
```

   The command output displays a list of available backup tar files.
   b. Select the backup tar file that you want to restore.

```
Choose backup option:1
Selected: EFA-2021.01.12-11.19.52.tar
Performing EFA restore using EFA-2021.01.12-11.19.52.tar
Generating backup before initiating restore
BACKUP_TAR: /apps/efa_logs/backup/EFA-2021.01.12-11.19.52.tar
Stopping all EFA services
All pods are terminated
Migrating database
Completed database migration
Checking if all PODS are in ready state...
```

```
Restore operation is successful
--- Time Elapsed: 9m3.079104969s ---
```

c. When the restore is complete, run **source/etc/profile**.

You can now log in to EFA.

Change the Host Name or IP Address

You can change the host name, the IP address, and the virtual IP address after EFA is deployed.

Host name changes are supported in single-node and multi-node deployments.

IP address changes are supported in single-node deployments.

Virtual IP address (VIP) changes are supported in multi-node deployments.

1. To change the host name, take the following steps.
   a. In a multi-node deployment, stop the monitoring service.

```
sudo systemctl stop efamonitor
```

   b. Change the host name of the system, such as with the following Linux command.

```
hostnamectl set-hostname <new name>
```

   c. Add the new host name to `/etc/hosts`.
   d. Run the following command as a root user or as a user with sudo privileges.

Important
Do not reboot the system before running this command.

```
$ sudo efa-change-hostname <old host name>

Reading host name of the system
Restarting mariadb service
Restarting k3s service
Checking k3s for the new host name
Host is in ready state in k3s
Setting current host as active node
Deleting old host name references
Waiting for EFA containers to start
Successfully updated host name in EFA
```

   In a single-node deployment, EFA is not operational during this step. In a multi-node deployment, EFA remains operational if the command is running on the standby node. EFA is not operational if the command is running on the active node.

   In a TPVM deployment, you can run the command from `/apps/bin/`.

   e. In a multi-node deployment, start the monitoring service.

```
sudo systemctl start efamonitor
```

2. To change the IP address of a single-node deployment, take the following steps.
   a. Run the following command as a root user or as a user with sudo privileges.

```
$ sudo efa-change-ip

Updating IP in EFA
Restarting k3s service
Updating all files with new IP
Deleting EFA services: gonotification-service gofabric-service gotenant-service
```

```
goauth-service gorbac-service goinventory-service goopenstack-service
govcenter-service gohyperv-service goraslog-service efa-api-docs gosystem-service
Waiting for EFA containers to start
Successfully updated IP in EFA
```

EFA is not operational during this step.

In a TPVM deployment, you can run the command from `/apps/bin/`.

b. After the IP address is updated, run `source /etc/profile` or open a new EFA session to log in.

3. To change the VIP of a multi-node deployment, take the following steps.

a. Stop the monitoring service.

```
sudo systemctl stop efamonitor
```

b. Run the following command as a root user or as a user with sudo privileges.

```
$ sudo efa-change-vip <new vip>

Updating all files with new VIP.
Restarting services on nodes
Waiting for EFA containers to start
Updating services with new VIP
Waiting for EFA containers to start
Successfully updated VIP for the installation.
```

EFA is not operational while this script runs.

In a TPVM deployment, you can run the command from `/apps/bin/`.

c. Start the monitoring service.

```
sudo systemctl start efamonitor
```

d. Run `source /etc/profile` or open a new EFA session to log in.

Display EFA Running Configurations

You can view the running-config of all current EFA configurations for core services.

The output is displayed in the following order: Asset, Fabric, Tenant commands. The command output contains the default values for each configuration line item.

You can use the command output for CLI playback on an empty EFA deployment, which is a useful tool for recovery.

Note

The output of **`efa show-running-config`** command is also captured as part of the supportsave zip file.

Run the **`efa show-running-config`** command.

```
$ efa show-running-config

efa inventory device register --ip "10.24.80.191" --username admin --password password

efa inventory device setting update --ip "10.24.80.191" --maint-mode-enable-on-reboot  No
--maint-mode-enable No --health-check-enable No --health-check-interval 6m
--health-check-heartbeat-miss-threshold 2 --config-backup-periodic-enable Yes
--config-backup-interval 24h --number-of-config-backups 4

efa inventory device register --ip "10.24.80.192" --username admin --password password
```

```
efa inventory device setting update --ip "10.24.80.192" --maint-mode-enable-on-reboot  No
--maint-mode-enable No --health-check-enable No --health-check-interval 6m
--health-check-heartbeat-miss-threshold 2 --config-backup-periodic-enable Yes
--config-backup-interval 24h --number-of-config-backups 4

efa fabric create --name "default" --type clos --stage 3 --description "Default Fabric"
```

This example shows only a partial list of typical output.

Audit Trail Logging

EFA provides full audit trail logging, including the successes and failures of user actions, which creates a 1-to-1 mapping between every action coming from EFA and a corresponding audit trail event from SLX.

Any configuration action on an SLX devices results in the generation of an audit trail. The name of the user is extracted from the token that the user logged in with. The user is assigned the role of `admin` as the default role on the device.

For OpenStack, the user name has the following format: `<OpenStack tenant UUID> - <OpenStack user name> - <EFA tenant name>`.

The following is an example of the audit log message for NETCONF or SSH sessions:

```
78 AUDIT, 2020/01/26-14:04:21 (GMT), [DCM-1006], INFO, DCMCFG, <ClientUserID>/
<ClientRole>/10.6.46.51/SSH/netconf,, SLX, Event: database commit transaction, Status:
Succeeded, User command: "configure config username test1 role admin password ****".
```

The `ClientUserID` and `ClientRole` values are derived from the `User` and `AuditLogRole` variables, which originate from the values in the access token when the NETCONF or SSH session was established.

Transfer of Audit Trail Data

Audit trail data from SLX devices is transferred to EFA for delivery upstream using JSON structured data.

The data is transferred to an upstream web server at a predefined URL that is registered with EFA.

Incoming syslog messages from SLX to EFA are converted by a logging service on EFA into JSON data, as in the following example:

```
"message_id": "9999",
  "message": "Hello world",
  "source_ip": "192.168.10.1",
  "user": "admin",
  "severity": "INFO",
  "timestamp", "2020-02-11 19:23:58.383304",
  "extra_data": {}
}
```

EFA sends the messages by POST requests to an upstream web receiver.

Logging and Log Files

Log directories vary by deployment type and by application.

EFA logs are saved to the following locations:

- Non-TPVM deployments: `/var/log/efa`. The installation logs in the `/var/log/efa/installer` directory are a good source for discovering the reason for a failure.
- TPVM deployments: `/apps/efa_logs`
- Kubernetes log files: `/var/log/pods`
- Keepalived service log files: `/supportsave/keepalived`. The directory contains Keepalived service logs and the `journalctl` log of Keepalived for the past day. This information is useful for helping debug failovers, double faults, and gateway connectivity.

In multi-node, high availability deployments, logs are replicated on all nodes in the cluster.

The **`efa system supportsave`** script gathers all logs, database dumps, pod logs, and deployment details and then compresses them into a ZIP folder. You can share this ZIP folder with Extreme support personnel when troubleshooting an issue.

Data Consistency

EFA ensures that SLX devices have the correct configuration before allowing traffic.

Overview

EFA is the data owner and Single Source of Truth (SSOT) for fabric configuration. The following figure describes how data is rendered consistent among EFA services.

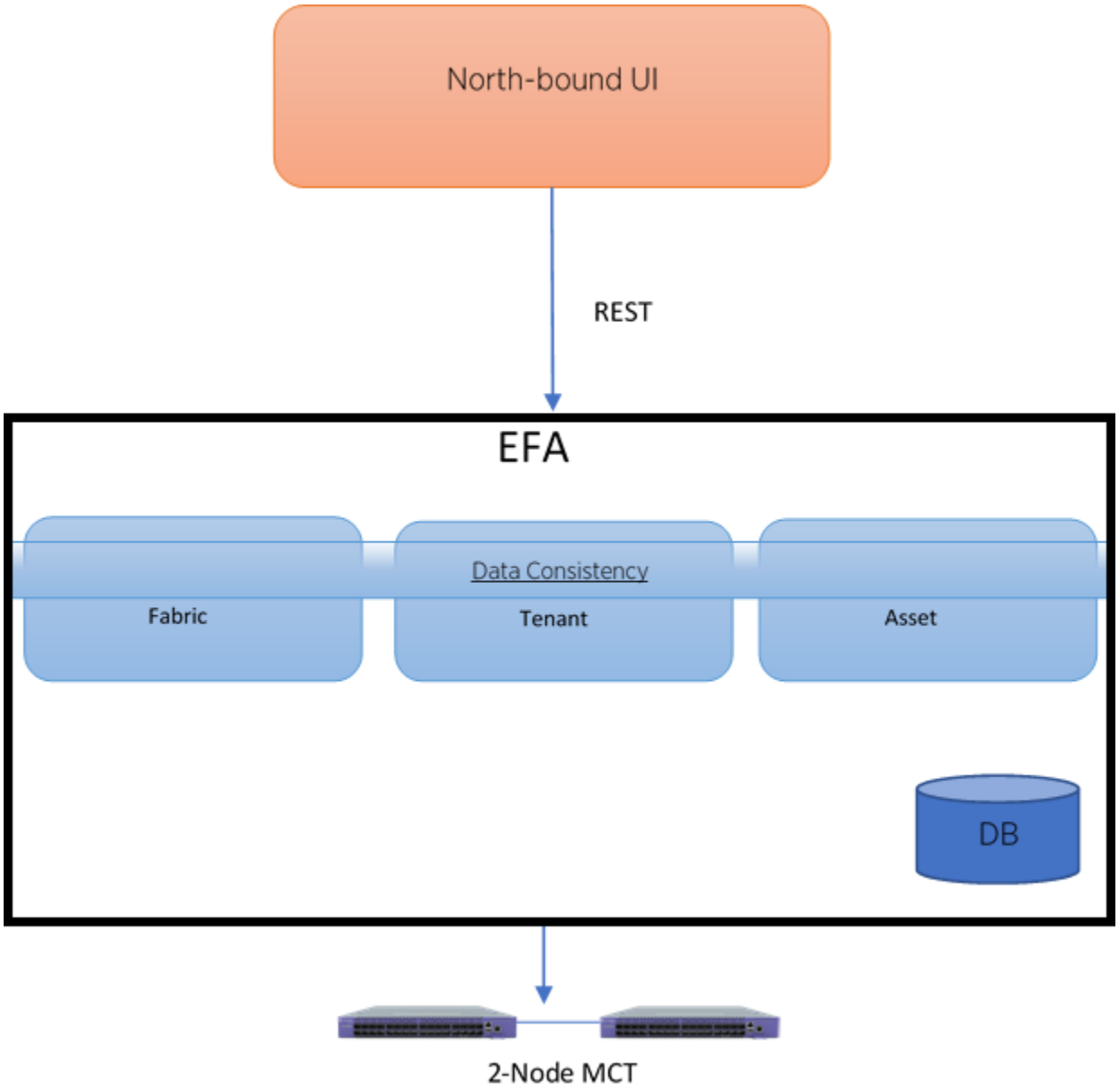


Figure 4: Data consistency overview

North-bound applications invoke REST APIs to perform various operations on EFA. EFA ensures that the operations leave EFA and the fabric in a consistent state.

Limitations

- You cannot use the SLX CLI to configure the entities that are managed by EFA.
- EFA can reconcile only those entities or configurations that it manages.
- EFA cannot modify out-of-band entities or configurations unless they conflict with the configurations that it manages.

Periodic Device Discovery

Tenant and Fabric Services use periodic discovery to detect out-of-sync configurations on the devices. These Services act on the published events and update the database to reflect the status of the devices as in-sync and out-of-sync.

You can perform on-demand device discovery using the command line.

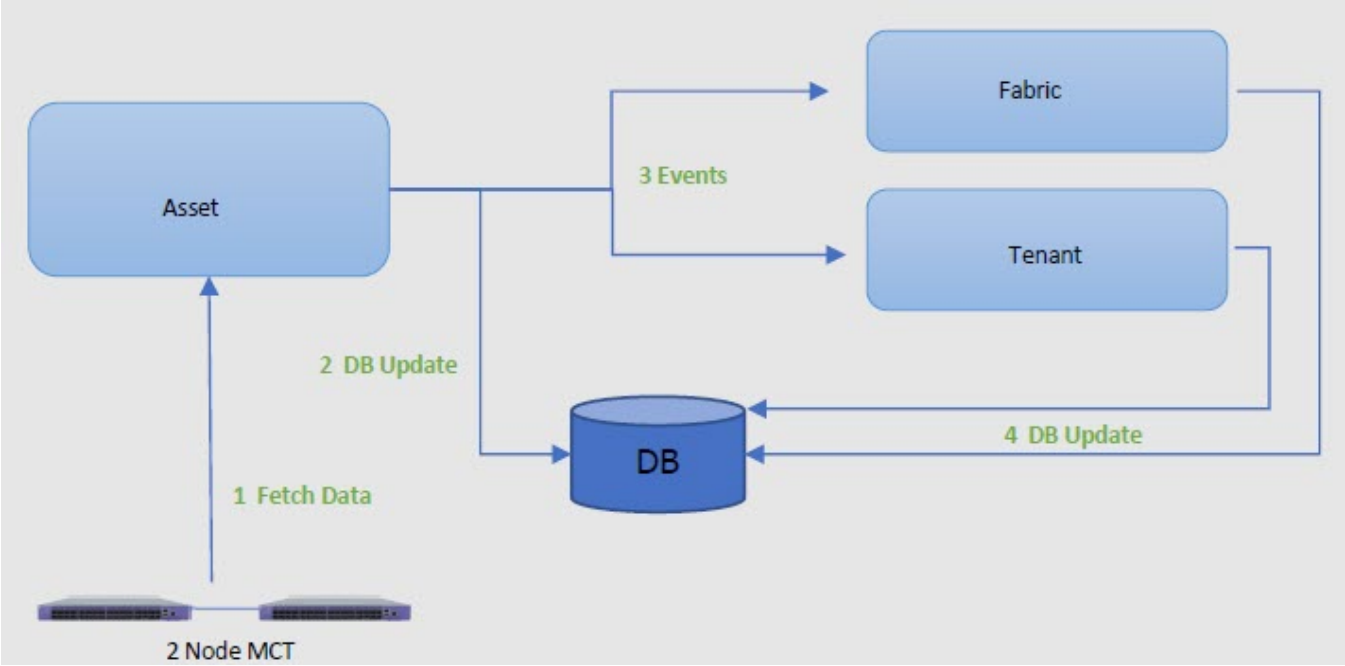


Figure 5: Device discovery and database updates

The Asset service periodically polls the devices in the fabric and keeps the database and other services updated of any changes in the underlying fabric. The default polling interval is one hour, with valid values ranging from 15 minutes to 24 hours.

You can use the **`efa inventory device discovery-time list`** command to view the current discovery interval for a device or fabric. You can use the **`efa inventory device discovery-time update`** command to configure the discovery interval.

Persistent Configuration

Extreme devices support three types of configuration files:

- Default - Default configuration files are part of the firmware package for the device and are automatically applied to the startup configuration.
- Startup - Startup configuration files are persistent and are applied after system reboot.
- Running - Configuration currently effective on the device is the running configuration.

For more information on configuration files, see Extreme SLX-OS Management Configuration Guide.

In SLX-OS 20.1.1, the configuration management process maintains two databases, Running and Startup.

In SLX-OS 20.1.2 and later, all the configurations are stored in one database, which also persists.

- The **show running-config** command fetches the configuration from the database.
- The **copy running-config startup-config** command creates or updates the persistent configuration.
- After a upgrade or downgrade, replaying the startup file resumes the SLX database cleanup operations.

Maintenance Mode

In SLX-OS 20.1.1, maintenance mode can be enabled by configuring **enable** under system-maintenance configuration mode. If the configuration is persistent, the switch needs to be in maintenance mode before rebooting for it to come back in maintenance mode.

In SLX-OS 20.2.1 and later, maintenance mode can be enabled by configuring **enable-on-reboot** under system-maintenance configuration mode. After the reboot, the device comes back up in maintenance mode and remains operational.

```
SLX(config-system-maintenance)# enable-on-reboot
SLX(config-system-maintenance)# [no] enable-on-reboot
```

The **system maintenance turn-off** command brings the system out of maintenance mode.

Non-Reachable Devices

EFA tracks devices by running heart-beats to the SLX devices.

When a non-reachable device becomes reachable, EFA identifies any drift and performs reconciliation, if necessary.

Drift and Reconcile

EFA provides APIs to initiate drift and reconcile requests. Drift and reconcile support is provided at the device level. The unit of comparison is a single device whose configuration is compared with EFA and reconciled in case of a drift in the configuration.

Drift and reconcile is used during the following operations:

- Switch replacement
- After the reboot of a device in maintenance mode

Drift and Reconciliation Engine

The APIs for Drift and Reconcile perform the following operations:

Note
If **maintenance-mode-enable on reboot** is not set on the devices, Data Consistency is not guaranteed and Drift And Reconciliation operation is skipped.

1. Raslog received from the switch starts the state engine for reconciliation of the device.
   a. Initiate reconciliation of the Fabric Service
   b. Initiate reconciliation of the Tenant Service

2. Identify the drift in configuration by comparing the fabric configurations in Fabric Service with configurations in Asset service. Fabric Service performs reconciliation and pushes the intended configuration from fabric to the device.
3. Identify the drift in configuration by comparing the Tenant configurations in Tenant Service with configurations in Asset service. Tenant service performs reconciliation and pushes the intended configuration from fabric to the device.

The reconcileAPI does not perform reconciliation on the device. The reconcileAPI only identifies the configuration drift and displays the information. This API can also initiate device discovery before starting the reconcile engine.

To improve performance, the drift computation is done in multiple go-routines and bulk switch configurations per device as applicable.

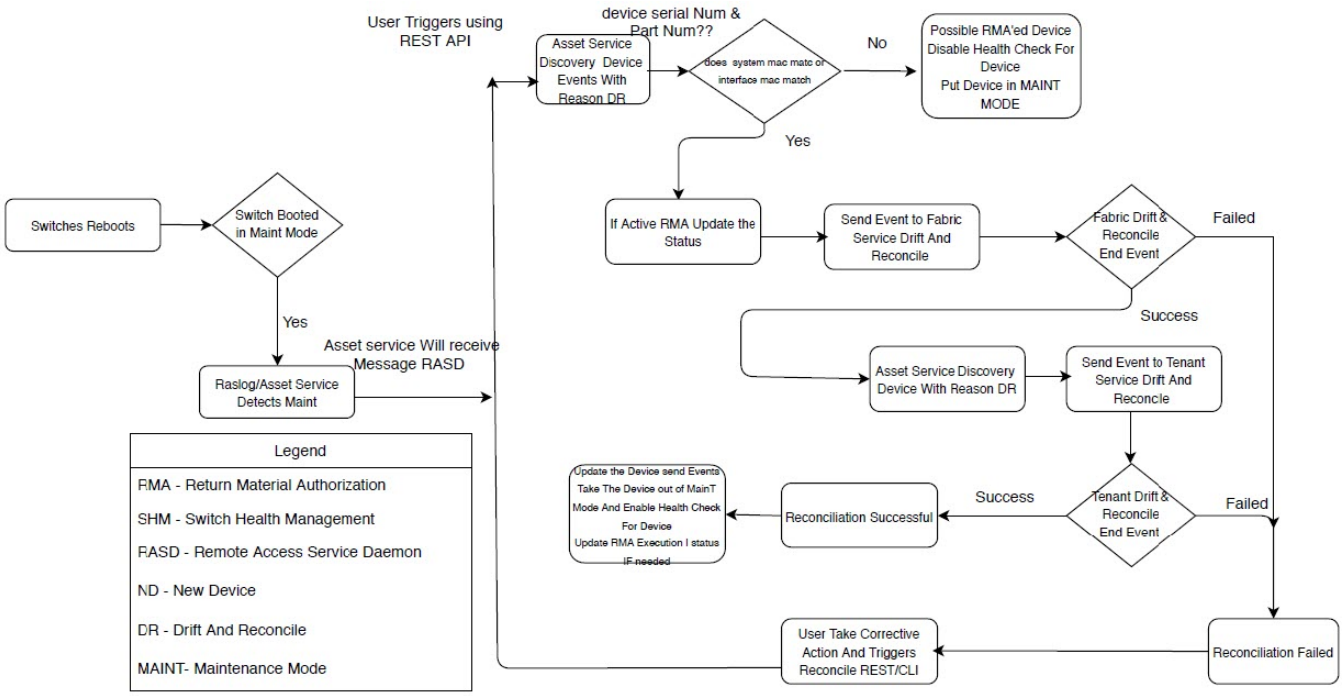


Figure 6: Drift and reconcile workflow

Drift and Reconcile Limitations in Tenant Service

Drift and Reconcile must be triggered after Inventory Service completes the event handling which are generated by Tenant-Service epg operations.

The following VLAN based EPG example shows how events are handled by the Inventory Service.

1. Create an VLAN based EPG on the device, D1 with `ctag-range 100`.
2. After successful epg creation, events containing new `epg-config` are published to the message bus.

Note

If the vlan(100) is manually deleted and the Inventory service cannot find the vlan(100) on the device D1, delete event for vlan(100) is not generated.

3. Inventory service processes the events, and updates the database through device discovery.

Idempotent Operations

The idempotent operations produce the same result for multiple identical requests or operations.

Reissuing an EFA command should leave the system in the same state as the last time the command was run. Such idempotent operations help ensure data consistency during high-availability failovers.

The following EFA fabric and tenant commands or operations are idempotent:

- fabric setting update
  - optimized-replication-enable
  - mdtgroup-range
  - default-mdtgroup
- tenant create
- tenant update
  - desc-update
  - vni-update
  - port-add
  - port-delete
  - vlan-add
  - vlan-delete
  - vlan-update
  - num-vrf-update
  - enabled-bd-update
- tenant delete
- end point group create
- end point group update
  - port-group-add
  - port-group-delete
  - ctag-range-add
  - ctag-range-delete
  - vrf-add
  - vrf-delete
- end point group delete
- portchannel create
- portchannel update
  - port-add
  - port-delete
- portchannel delete
- vrf create
- vrf delete

In this example, running the **efa fabric create** command twice, with the same parameters, produces the same result each time.

```
$ efa fabric create --name fabric1  --type non-clos --description non-clos-fabric
Create Fabric nonclos [Success]

(efa:extreme)extreme@tpvm:~$ efa fabric create --name fabric1 --type non-clos --
description non-clos-fabric
Create Fabric nonclos [Success]
```

Rollback Scenarios for Data Consistency

Rollback of failed configuration changes ensures data consistency.

Failure on Some Devices During Configuration

When a REST operation succeeds on one device but fails on another, configuration changes are rolled back for both devices. In the following example, the operation fails on one MCT node but succeeds on the other. The whole operation fails and an error message is returned as part of the REST response.

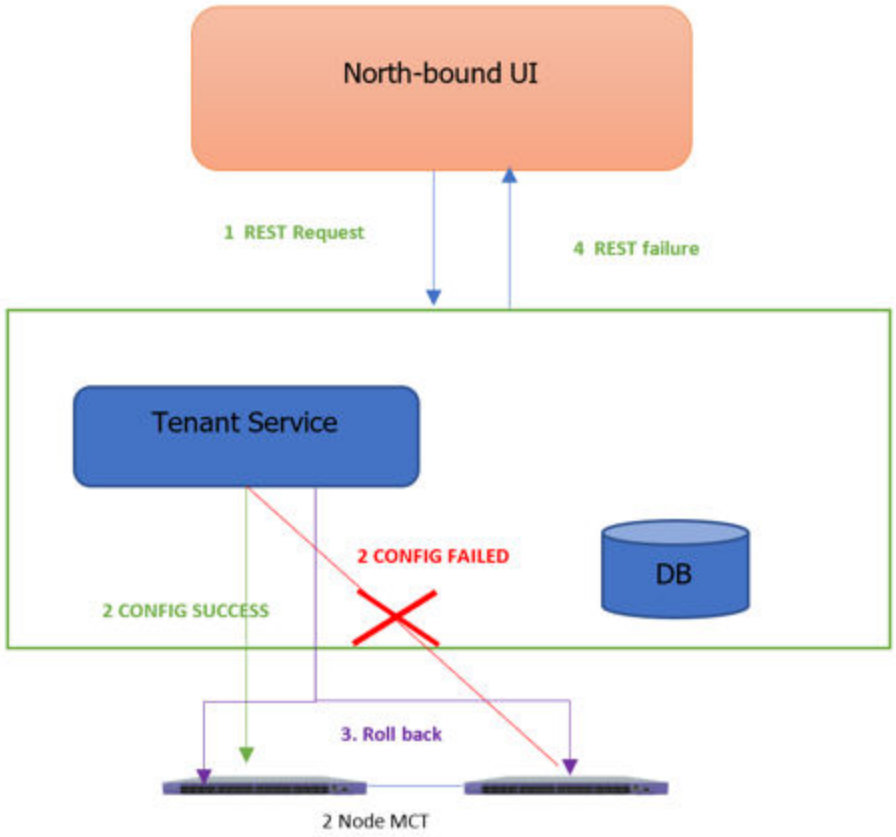


Figure 7: Rollback for failure of one node

Note

This process for partial failures is the default. You can change the process to enable partial successes even when one node fails. For more information, see Administered Partial Success on page 125.

Failure on All Devices During Configuration

When a REST operation fails on all devices in the request, configuration changes are rolled back for all devices. In this example, the operation fails on both MCT nodes and an error message is returned as part of the REST response.

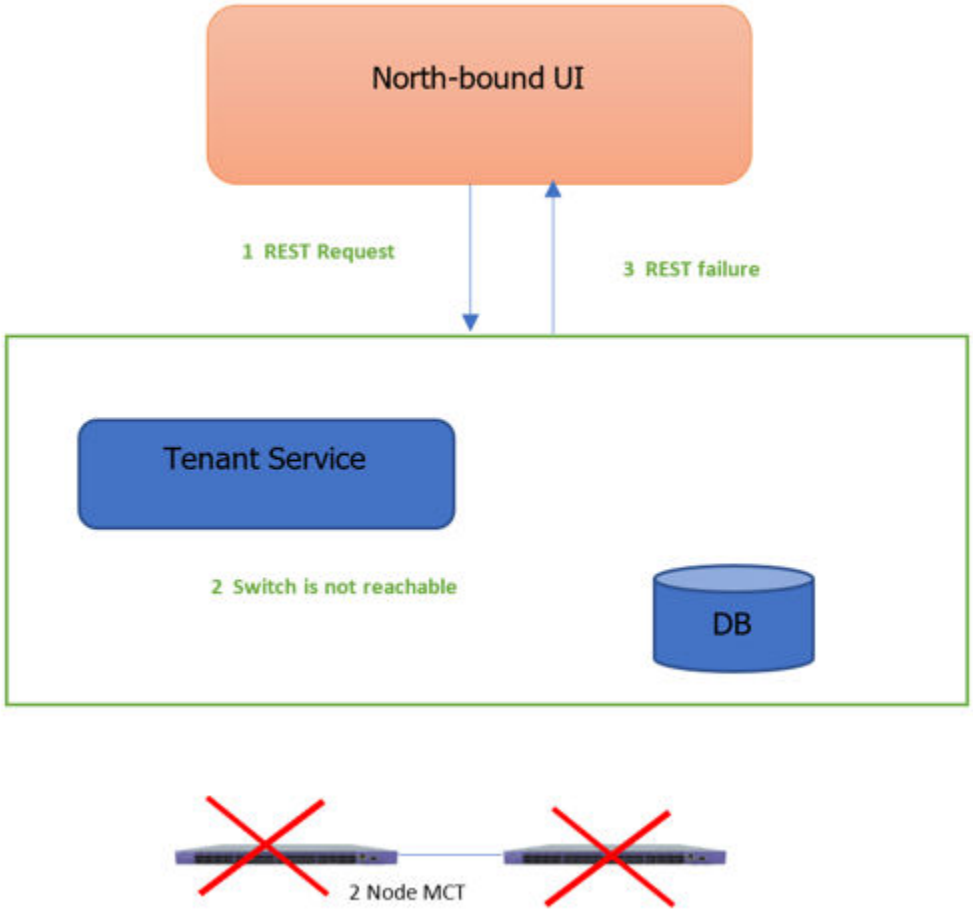


Figure 8: Rollback for failure of both nodes

Failure During De-configuration

Rollback does not occur when a REST operation fails during a de-configuration request. The status of configuration items that were not rolled back changes to "delete-pending." You must manually verify and address the status of such items.

EFA High Availability Failover Scenarios

EFA high availability provides for uninterrupted service in several different scenarios.

For information about deploying EFA for high availability, see the Extreme Fabric Automation Deployment Guide, 2.3.0.

SLX device failure

When an SLX device fails, the SLX-OS and the EFA services running on TPVM go down for the failed node. The time it takes for failover to the standby node varies depending on whether the K3s agent node is actively running the EFA services. The following image depicts a scenario in which one SLX device fails.

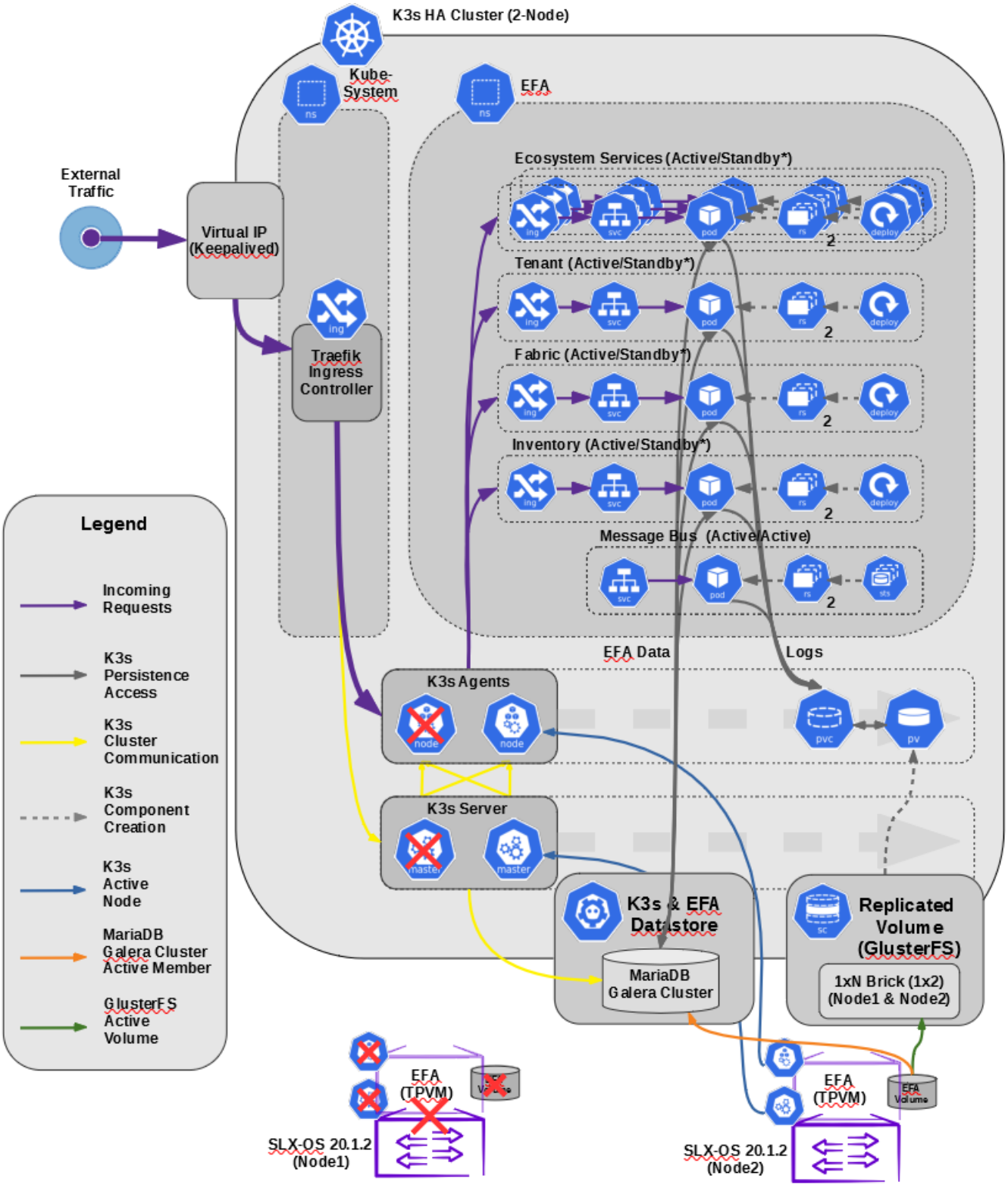


Figure 9: SLX device failure in a two-node cluster

SLX device failure on the active K3s agent node

When the K3s agent node is actively running EFA services on a node that fails, K3s initiates failover and starts the EFA services on the standby node. Failover is complete when EFA services are running on the newly active K3s agent node (node 2).

Because the GlusterFS replicated volume remains available during failover, the K3s cluster data store and the EFA data store remain operational.

When the failed node is again operational, it becomes the standby node. The K3s agent node continues to run EFA services from node 2. When both nodes are up and K3s is running, all services fetch the latest data from devices to ensure that EFA has the latest configurations.

SLX device failure on the standby K3s agent node

When the K3s agent node is the standby and is not running EFA services, no failover actions occur if this node fails. EFA services continue to run on the active node without interruption.

TPVM failure

The TPVM failure scenario is similar to that of the SLX device failure scenario. The only difference is that SLX-OS continues to operate.

Two-node failure

In the unlikely event that both nodes in the cluster fail at the same time (for reasons such as a power failure or the simultaneous reboot of SLX devices), EFA has built-in recovery functionality. If the cluster is not automatically recovered within 10 minutes of power being restored or within 10 minutes of the TPVM being rebooted, then you can manually recover the cluster.

For more information, see #unique_42.

Multiple Management IP Network

EFA is supported on multiple management IP networks, providing access to EFA from servers that are on different subnets.

Overview

The Multiple Management IP Network feature offers the following support:

- Supports single node and multi-node deployments
- Supports TPVM deployments and server and VM-based deployments
- You can add additional management networks during EFA installation and after EFA installation.
- You can delete management networks after EFA installation.
- The configuration of the multiple networks is migrated during all EFA upgrade scenarios: single node to single node, single node to multi-node, and multi-node to single node.

- If you do not need multiple management networks, simply reply "no" when prompted during EFA installation or upgrade. For instructions, see the installation and upgrade topics in the Extreme Fabric Automation Deployment Guide, 2.4.0.
- The feature supports up to 6 networks.
- The RMA, backup, restore, and upgrade functions are all supported with the multiple management IP feature.

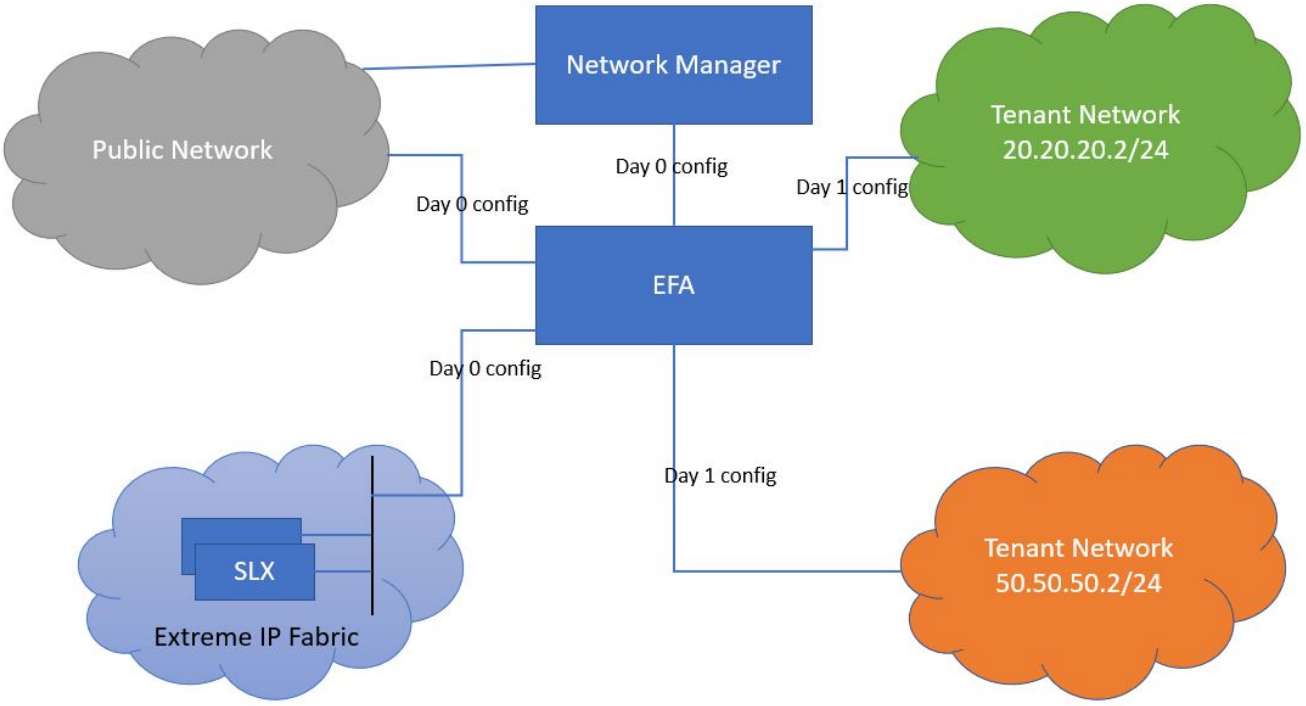


Figure 10: Multiple Management IP Network Diagram

Assumptions

- In a multi-node deployment, the sub-interface with the VLAN is created under the same NIC as the VIP destination. In a single-node deployment, the sub-interface is created under the NIC that you specified as the host IP installation (if there are multiple NICs). Creating sub-interfaces on different NICs of the server is not supported.
- EFA does not validate to the newer IP subnets. You are responsible for ensuring reachability.
- Changing IP subnets is not supported. To make any modifications to the IP subnet or VLAN or name, you must delete the IP subnet and add a new one.
- Because of the EFA-wide changes required by this feature (for example, updates to the Ingress controller), you can expect about 20 to 30 seconds of downtime during add and delete operations.
- In a high-availability deployment, both nodes have to be up and available during add and delete operations (because sub-interface creation and keepalived changes are unique to the node). Because this is an infrequent operation, you should verify that both the nodes are up and in READY state before beginning this operation.

Configuration Supporting Multiple Management IP Networks

The Multiple Management IP Network feature can scale up to 6 networks.

Third-party certificate changes

You can access EFA at different IP addresses (one for each of the new networks).

Update the third-party SSL certificates with the external IP address of management interfaces of EFA. Re-generate the certificates when you add a new network. You can replace the generated certificate with your own certificates (third-party certificates), which must have a reference to each of the EFA IP addresses.

Third party certificate must contain a Subject Alt Name (SAN) field for each EFA IP address. In particular, if you have added management access for external networks, include the EFA management IP address for each external network. The **`openssl`** command supports a flag for adding a SAN IP address.

For example:

```
-addext "subjectAltName = IP.1:192.168.30.40"
```

Day 0 and installation changes

- In a high-availability deployment, the VIP (virtual IP address) that you enter as part of installation remains the same. This VIP is distinguished from those added during Multiple Management IP Network operations and cannot be deleted.
- During installation, you are prompted to create additional Multiple Management IP Networks.
- Once you have specified all the IP address and VLAN combinations, installation proceeds as with earlier releases of EFA.
- Keepalived, ingress, and interface changes are done as part of installation on both the nodes of a high-availability deployment.
- Configuration is persisted for RMA purposes, so that the Supportsave function has data for debugging issues.

Day 1 to n changes

- You can add and delete IP address and VLAN combinations after installation using the EFA CLI or the REST APIs.
- Keepalived, ingress, and interface changes are done as part of this operation on both the nodes.
- Configuration is persisted for RMA purposes, so that the Supportsave function has data for debugging issues.
- The backup and restore process also restores the previous configuration of the sub-interfaces.

Installer changes

During installation, you are asked whether you want to add additional management networks for connection to EFA. If you select `Yes`, you are then asked to provide three input parameters:

- Sub-interface name, which is a unique name that contains no more than 11 characters, no white space, and no **%** or **/** characters.
- ID of the VLAN that the management network uses to tag traffic. Valid values range from 2 through 4093.

- IP subnet address in CIDR format. The subnet must not overlap with any IP subnet that you have already provided.

You repeat this process until you have finished adding all the sub-interface information you need. Then you select No to continue with installation. For details, see the installation and upgrade information in the Extreme Fabric Automation Deployment Guide, 2.4.0.

Adding and Deleting Management Subinterfaces

You can use the EFA CLI to add and delete management sub-interfaces.

For more information about the following commands, see the Extreme Fabric Automation Command Reference, 2.4.0.

Syntax to add a management sub-interface

efa mgmt subinterface create --name <name> --ip-address <IP Subnet> --vlan-id <VLAN>

If a management network with the same name exists, the users are informed about it and this operation fails.

The changes made by this operation span across three different components:

- Sub-interface creation under the physical NIC
- Keepalived configuration changes (for high-availability deployments)
- Ingress controller changes

If any of the operations to the component fails, it is marked as a failed operation and the configurations return to the previous state.

Syntax to delete a management sub-interface

efa mgmt subinterface delete --name <name>

If a management network with the name exists, it is deleted. Otherwise, the correct response is provided in the command output.

Complete syntax examples

```
$ efa mgmt subinterface?
Management subinterface commands

Usage:
  efa mgmt subinterface [command]

Available Commands:
  create        Create sub-interface (sub-interface)
  delete        Delete sub-interface (sub-interface)
  show          List of sub-interfaces (sub-interfaces)

Flags:
  -h, --help          help for sub-interface
```

```
Use "efa mgmt subinterface [command] --help" for more information about a command.
```

```
$ efa mgmt subinterface create -h
Create management subinterface (sub-interface)
Usage:
  efa mgmt subinterface create [flags]
Flags:
      --name string         Name of the sub-interface
      --vlan-id int         VLAN Id of sub-interface
      --ip-address string   IP Address of sub-interface including subnet mask.
                            Example: 10.24.80.150/24
  -h, --help                help for create
```

```
$ efa subinterface delete -h
Delete management subinterface (sub-interface)
Usage:
  efa mgmt subinterface delete [flags]
Flags:
      --name string   Name of the sub-interface
  -h, --help          help for delete
```

```
$ efa mgmt subinterface show -h
List of management sub-interfaces (sub-interfaces)

Usage:
  efa mgmt subinterface show [flags]
 Flags:
       --name string   Name of the sub-interface
   -h, --help          help for show
```

```
$ efa mgmt subinterface create --name server1 --vlan-id 20 --ip-address
          20.20.20.2/24
Subinterface server1 created successfully
```

```
$ efa mgmt subinterface delete --name server1
Subinterface server1 deleted successfully
```

```
$ efa mgmt subinterface show
+---------------+------------------+------+---------------+
| Sub-Interface | Parent Interface | Vlan |   IP Subnet   |
+---------------+------------------+------+---------------+
| server1       | eth0             | 20   | 20.20.20.2/24 |
+---------------+------------------+------+---------------+
| server2       | eth0             | 50   | 50.50.50.2/24 |
+---------------+------------------+------+---------------+
Management Subinterfaces Details
```

```
$ efa mgmt subinterface show --name server1
+---------------+------------------+------+---------------+
| Sub-Interface | Parent Interface | Vlan |   IP Subnet   |
+---------------+------------------+------+---------------+
| server1    | eth0             | 20   | 20.20.20.2/24 |
+---------------+------------------+------+---------------+
Management Subinterface Details
```

Example configurations

This topic provides an example of configurations that this feature modifies to ensure the appropriate verification of the changes.

Sub-Interface creation

Create a new VLAN interface on each node of the EFA install. The following example shows the creation of a sub-interface with the name “sub1”. It is a child interface of the default management interface, “eth0”. EFA adds a prefix “efa-“ to the VLAN interface name, meaning it is instantiated as “efa-sub1”. EFA manages these interfaces, and does not rely on Linux network managers such as NetworkManager and systemd-networkd (no files are changed under `/etc/network/interfaces.d`).

```
$ ip addr show type vlan
25: efa-sub2@eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UP group default qlen 1000
    link/ether 52:54:00:90:b3:da brd ff:ff:ff:ff:ff:ff
    inet 20.20.20.2/24 scope global efa-sub2
       valid_lft forever preferred_lft forever
    inet6 fe80::5054:ff:fe90:b3da/64 scope link
       valid_lft forever preferred_lft forever
26: efa-sub1@eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UP group default qlen 1000
    link/ether 52:54:00:90:b3:da brd ff:ff:ff:ff:ff:ff
    inet 50.50.50.2/24 scope global efa-sub1
       valid_lft forever preferred_lft forever
    inet6 fe80::5054:ff:fe90:b3da/64 scope link
       valid_lft forever preferred_lft forever
```

Use the following command to list all the VLANs:

`$ ip addr show type vlan`

Keepalived Configuration

Once the user configures a new VLAN/sub-interface through EFA on both the active and standby nodes, the sub-interfaces are created.

On the active node the IP subnet is applied to the sub-interface.

The keepalived configuration with the new sub-interface IP subnets are updated on both active and standby nodes

Keepalived runs only one instance of VRRP and multiple VIPs can be assigned.

```
vrrp_instance HA1 {
  virtual_router_id 51
  advert_int 1
  priority 102
  nopreempt
  interface eth0
  unicast_src_ip 10.24.95.134    # IP address of Master Server
  unicast_peer {
    10.24.95.69                # IP address of Slave Server
  }
  virtual_ipaddress {
    10.24.95.116/32 dev eth0
    20.20.20.2/24   dev efa-sub2
    50.50.50.2/24   dev efa-sub1
  }
  track_script {
    chk_default_gw
    chk_mariadb
  }
```

```
  notify /apps/bin/keepalivednotify.sh
}
```

The first VIP is provided in all HA installations and is not managed by MMIP CLIs. In particular, it can't be removed by MMIP. On failover, EFA configures the new active node sub-interfaces with the IP subnets. This is achieved through Keepalived which runs the VRRP protocol

Keepalived runs only one instance of VRRP and multiple VIPs can be assigned.

Ingress Controller Configuration

Currently, a single IP used as an external IP from the ingress controller in HA environment. This is enhanced to have multiple external IPs from EFA 2.4.0.

Verify that the IP addresses are associated to the ingress controller by executing the following:

```
$ k3s kubectl get svc traefik -n kube-system

NAME          TYPE           CLUSTER-IP      EXTERNAL-IP
PORT(S)                          AGE

traefik   LoadBalancer   10.43.214.249
10.175.100.240,10.175.100.186,10.175.100.117,20.20.20.2,50.50.50.2
80:31135/TCP,443:30102/TCP   105m
```

The output must have all the management network IPs listed there. The other way to verify that all the IP addresses are bound to the ingress is to verify the output of the following command.

`netstat -tuplen | grep :443`

EFA's default SSL certificate is also updated by an Add or Delete MMIP operation. This is necessary because EFA's ingress controller must be able to select the correct certificate to authenticate the incoming connection. This is achieved by comparing each of the controller certificate's Subject Alternate Names (SANs) fields with the IP of the incoming connection (because this is TLS, and it happens logically before the L7 connection is established). EFA regenerates an SSL certificate and ensure that all MMIP IP addresses are listed as SANs. The users can confirm this with the following command:

```
$ openssl x509 -text -noout -in /apps/efadata/certs/own/tls.crt | grep -A1
"Subject Alternative Name"

            X509v3 Subject Alternative Name:

                DNS:efa.extremenetworks.com, IP Address:127.0.0.1, IP
Address:10.175.100.240, IP Address:10.175.100.186, IP Address:10.175.100.117, IP
Address:20.20.20.2, IP Address:50.50.50.2
```

To confirm that Traefik, the ingress controller is using this cert, run the following:

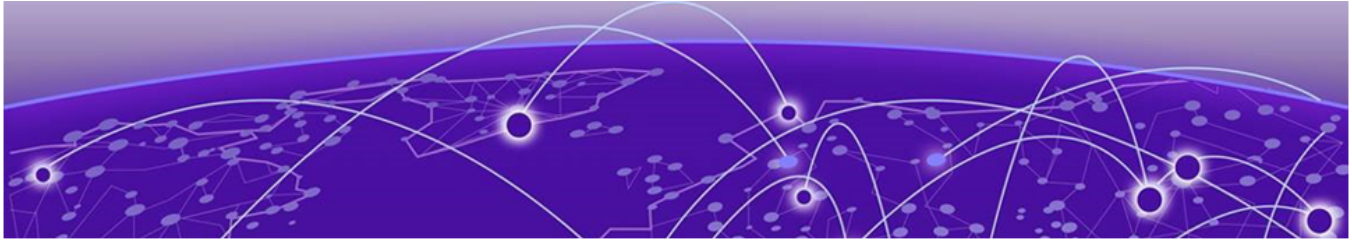
```
$ k3s kubectl get deployment traefik -n kube-system -o yaml | grep
'secretName: efasecret'

              secretName: efasecret-tls
```

The secret must be called "efasecret-tls". To confirm the secret is correct, run the following:

```
k3s kubectl get secret efasecret-tls -n kube-system -o yaml
```

Extract the field **tls.key** from the embedded JSON output and base64-decoded. This results in a certificate which can be verified with the above openssl command.

Note that if the users provide their own certificates (third-party certificates) for the use within EFA and wishes to make use of the MMIP feature, they must ensure that the certificate contains SAN fields for each of the MMIP interfaces and the distinguished VIP interface.



Fabric Infrastructure Provisioning

Fabric Service Overview

Fabric Service is responsible for automating the Fabric BGP underlay and EVPN overlay. By default, the EVPN overlay is enabled but can be disabled before provisioning if desired. Fabric Service exposes the CLI and REST API to clients for automating the fabric underlay and overlay configuration.

Fabric Service features include:

- Small Data Center Topology (non-Clos support)
- Support for 3- and 5-stage Clos fabrics
- Support for MCT configuration
- Support for Eco-System Integration; Openstack, VMWare vCenter, Microsoft Hyper-V/SCVMM

Underlay automation includes Interface Configurations (IP Numbered), BGP Underlay for spine and leaf, BFD, and MCT configurations. Overlay automation includes EVPN and Overlay Gateway configuration. Fabric Service is deployed along with Inventory Service and Tenant Service.

IP Fabric and Clos Orchestration Overview

A fabric is a logical container for holding a group of devices. Here it denotes a collection of devices that are connected in a fabric topology and on which you can configure underlay and overlay.

Fabric service provides following features:

- 3-stage Clos automation
- 5-stage Clos automation

- Small Data Center automation
- Multi-Fabric automation
- Fabric topology view
- Fabric validation, error reporting, and recovery
- Single-homed leaf or multi-homed (MCT) leaf

Fabric CLIs and REST APIs provide the following:

- Mechanism to create a fabric composed of multiple DC points of delivery (PoDs).
- Mechanism to configure fabric settings. Fabric settings are collections of settings that control the various parameters of the fabric being managed, for example, Layer 2 and Layer 3 MTU, and BGP maximum paths.
- Mechanism to fetch per-device errors occurring during fabric configuration, for which you can take corrective or remedial actions.

Errors occurring on the device during fabric creation are tagged against the devices and can be retrieved from the CLI and REST APIs for use in taking corrective or remedial actions.

SLX Device Prerequisites for Fabric Service

The following items are required before you configure your fabric.

- Management IP addresses must be configured on all devices.
- SLX devices must have the appropriate firmware version. For more information, see the list of supported platforms in the Extreme Fabric Automation Deployment Guide, 2.4.0.
- SLX 9850: Fabric links must be enabled manually, through `no shut`.
- SLX 9540: The appropriate TCAM profile must be set and the device rebooted.

```
device# conf
Entering configuration mode terminal
device(config)# hardware
device(config-hardware)# profile tcam vxlan-ext
%Warning: To activate the new profile config, run 'copy running-config startup-config'
followed by 'reload system'.
device(config-hardware)#
```

- Refer to the release-specific *Extreme SLX-OS Management Configuration Guide* for configuration steps for each platform.

Clos Overview

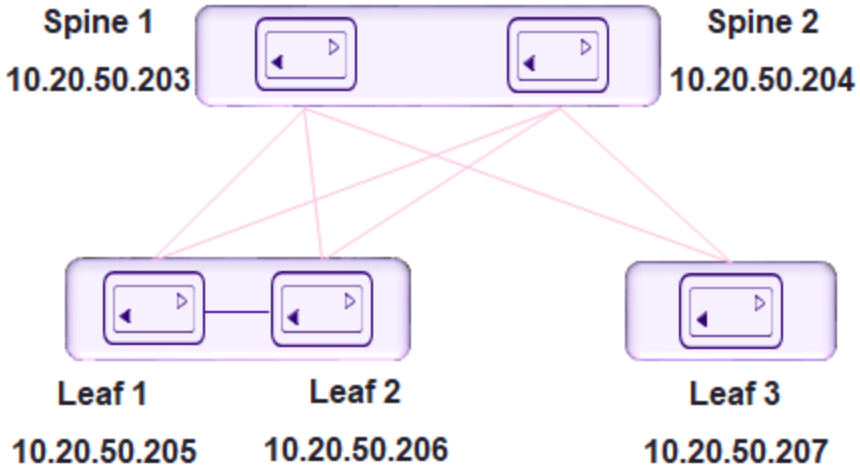
Extreme Fabric Automation (EFA) manages the life cycle of IP Fabric Clos networks.

EFA offers unique flexibility in supporting 3- and 5-stage Fabric Clos topologies based on a BGP underlay with a BGP or EVPN overlay.

Tenant Network onboarding services are supported on both topologies, allowing you to create connectivity for devices connected to the fabric, such as compute (servers), storage, and connectivity to external routers or gateways.

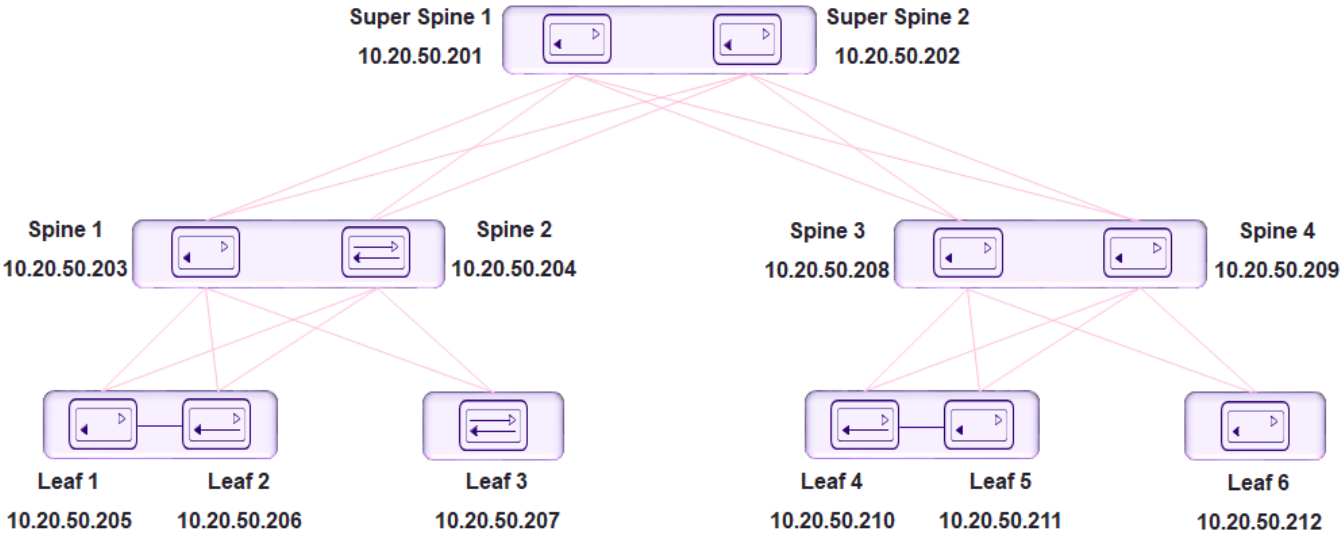
3-stage Clos

3-stage Clos consists of an ingress leaf layer, a middle spine layer, and an egress leaf layer. Servers are connected to leaf devices and leaf devices are connected to all spines. No leaf devices are connected to other leaf devices, nor are spines connected to spines. Data enters at an ingress leaf, is routed through a spine to an egress leaf, and then out of the network to the next server in the path. In this topology, servers are always 3 hops (leaf, spine, leaf) away from another server.



5-stage Clos

5-stage Clos is a 3-stage topology that is divided into clusters and on which a Super-spine layer is added. All links between leaf and spine must be connected. Spine are not be interconnected. Similarly, all the links between the spine and Super-spine must be connected.



Configure a 3-Stage Clos Fabric

The 3-stage topology has 2 layers of devices: leaf and spine. All links between leaf and spine must be connected. Spine nodes are not interconnected.

1. Create the fabric.

```
efa fabric create
```

2. Add a device to the fabric.

```
efa fabric device add
```

A device must be registered with the Inventory Service before you can add it to a fabric. However, if you provide a user name and password when you run the command, then the devices are automatically registered with the Inventory Service. See the examples at the end of this procedure.

You can add multiple devices by using the `efa fabric device add-bulk` command.

Tip

To validate fabric port-link status, complete the following operations before **`efa fabric device add-bulk`** command:

a. Run the **`efa inventory device register –ip <list of device-ips>`** command.
b. Run the **`efa inventory device interface list –ip <device-ip>`** command.
   i. Verify port link status (up/down) in Admin Status and Oper Status fields.
   ii. Confirm they are as expected.
   iii. If not, manually check for physical cabling and fix any issues. Continue with the `efa fabric device add-bulk` operation.

3. Configure the fabric.

```
efa fabric configure
```

Topology validation occurs during the addition of a device and during fabric configuration. The following validations are performed.

- Leaf nodes must connect to all the spine nodes.
- A spine node must connect to all the leaf nodes.
- A Border leaf node connects to all the spine nodes.
- A spine node connects to all the Border Leaf .
- No more than two leaf nodes connect to each other.
- No more than two Border Leaf nodes connect to each other.
- Border Leaf node and leaf node are not connected to each other.
- Spine nodes are not connected to each other.
- Super-spine nodes are not connected to each other.
- A leaf node marked as "multi-homed" must have an MCT neighbor.
- A leaf node marked as "single-homed" is not connected to other leaf nodes.
- A Border Leaf node marked as "multi-homed" must have an MCT neighbor.

- A Border Leaf node marked as "single-homed" is not connected to other Border Leaf nodes.
- Device role (such as Leaf, Border-leaf, Spine, or Super-spine) is validated for a given device platform type (for example, SLX 9840 cannot be added as a leaf).

Tip

The validation process reports any errors as a response to the **efa fabric device add** or **efa fabric configure** operations. You can use the **efa fabric error show** command to export these errors to a CSV file.

Note

You cannot change fabric settings after you add devices to the fabric.

This example creates the fabric.

```
efa fabric create --name stage3
```

This example adds multiple devices to the fabric.

```
efa fabric device add-bulk --leaf 10.20.50.205,10.20.50.206,10.20.50.207
--spine 10.20.50.203,10.20.50.204 --name stage3 --username admin
--password password
```

This example configures the fabric.

```
efa fabric configure --name stage3
```

Configure a 5-Stage Clos Fabric

The 5-stage topology has 3 layers of devices: leaf, spine, and super-spine.

You can build a 5-stage Clos from top to bottom or bottom to top. The following example builds from top to bottom.

1. Create the fabric.

   ```
   efa fabric create
   ```

2. Add a device to the fabric.

   ```
   efa fabric device add
   ```

   A device must be registered with Inventory Service before you can add it to a fabric. However, if you provide a user name and password when you run the command, then the devices are automatically registered with the Inventory Service. See the examples at the end of this procedure.

   You can add multiple devices by using the `efa fabric device add-bulk` command. If you choose to add multiple devices in bulk, ensure you perform the following operations first:

   - Run the **efa inventory device register --ip <list-of-device-ips>** command.
   - Run the **efa inventory device interface list --ip <device-ip>** command. In the output of the command, verify that the states of the port links are as you expected (in the Admin Status and Oper Status fields). If not, manually check the physical cabling and fix any issues. Then continue with the **efa fabric device add-bulk** operation.

3. Configure the fabric.

   ```
   efa fabric configure
   ```

Topology validation occurs during the addition of a device and during fabric configuration. The following validations are performed:

- Leaf nodes must connect to all the spine nodes.
- A spine node must connect to all the leaf nodes.
- A Border leaf node connects to all the spine nodes.
- A spine node connects to all the Border leaf nodes
- No more than two leaf nodes connect to each other.
- No more than two Border leaf nodes connect to each other.
- Border leaf node and leaf node are not connected to each other.
- Spine nodes are not connected to each other.
- Super-spine nodes are not connected to each other.
- A leaf node marked as "multi-homed" must have an MCT neighbor.
- A leaf node marked as "single-homed" is not connected to other leaf nodes.
- A Border leaf node marked as "multi-homed" must have an MCT neighbor.
- A Border leaf node marked as "single-homed" is not connected to other Border leaf nodes.
- Device role (such as leaf, Border-leaf, spine, and Super-spine) is validated for a given device platform type (for example, SLX 9840 cannot be added as a leaf).

Tip

The validation process reports any errors as a response to the **efa fabric device add** or **efa fabric configure** operations. You can use the **efa fabric error show** command to export these errors to a CSV file.

This example creates the fabric.

```
efa fabric create --name stage5
```

This example adds a device to the fabric.

```
efa fabric device add--name stage5 --username admin --password password
--leaf 10.20.50.205,10.20.50.206,10.20.50.207 --spine 10.20.50.203,10.20.50.204
--three-stage-pod podA --super-spine
```

This example adds multiple devices to the fabric.

```
efa fabric device add-bulk --name stage5 --username admin --password password
--leaf 10.20.50.205,10.20.50.206,10.20.50.207 --spine 10.20.50.203,10.20.50.204
--three-stage-pod podA --super-spine 10.20.50.201,10.20.50.202 --five-stage-pod podC
```

This example configures the fabric topology.

```
efa fabric configure --name stage5
```

Overview of Day-0 Operations for a Non-Clos Fabric

Day-0 operations consist of forming the fabric.

This table provides examples of the commands that you use to create a non-Clos fabric with two SLX devices. For more information about commands and supported parameters, see Extreme Fabric Automation Command Reference, 2.4.0.

Table 8: Day-0 operations

Operation	Command Example
Create a fabric	**efa fabric create --name CNCF type non-clos**
Select MCT ports	**efa fabric setting update --rack-ld-mct-ports '0/29,0/30,0/31,0/32' --name CNCF**
Enable backup routing	**efa fabric setting update --backup-routing-enable Yes --name CNCF**
Disable VLAN VNI auto-map	**efa fabric setting update --vni-auto-map No --name CNCF**
Add the first device	**efa fabric device add --ip 10.24.80.158 --hostname slx-a --rack pod1 --username admin --password password --name CNCF**
Add the second device	**efa fabric device add --ip 10.24.80.159 --hostname slx-b --rack pod1 --username admin --password password --name CNCF**
Configure the fabric	**efa fabric configure --name CNCF**

Supported Non-Clos Topologies

EFA supports non-Clos (small data center) fabrics.

EFA provides the following support for non-Clos fabrics (small data centers) on SLX 9140, SLX 9150, SLX 9250, SLX 9640 and SLX 9740 devices:

- Single rack automation. Each rack consists of a two-node MCT pair.
- Multi-rack automation
- Multi-homed leaf (MCT)
- Overlay-only automation
- Fabric topology view
- Fabric validation and troubleshooting

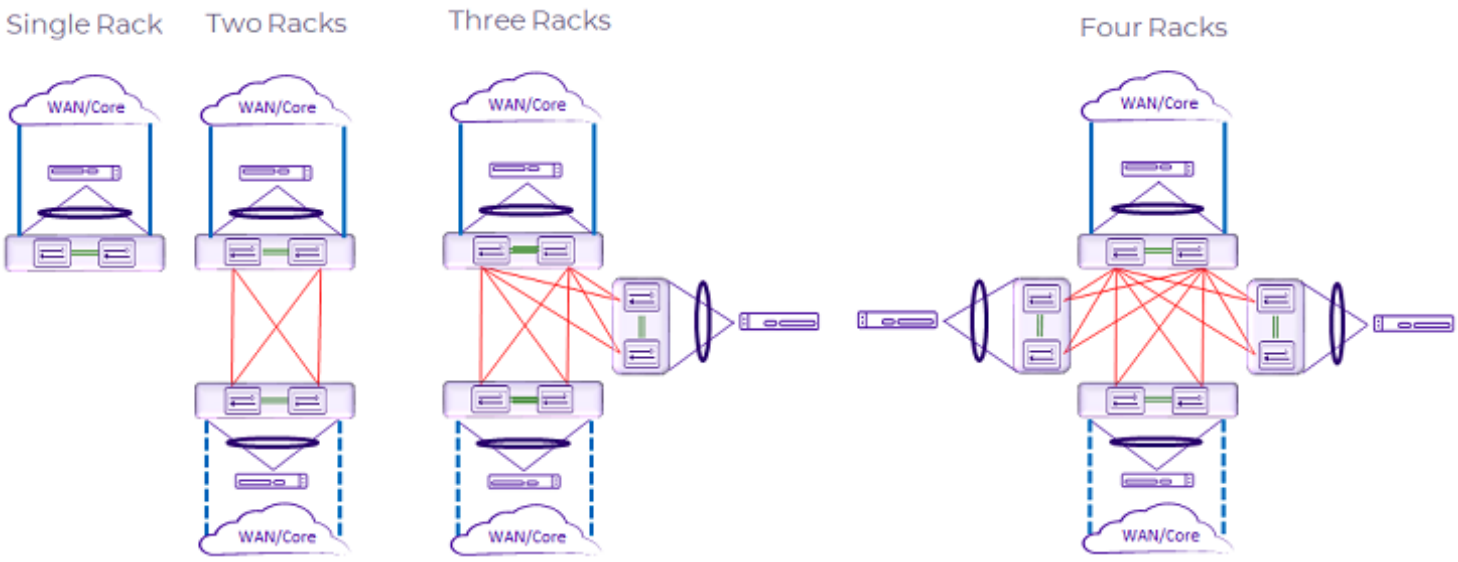


Figure 11: Supported small data center topologies

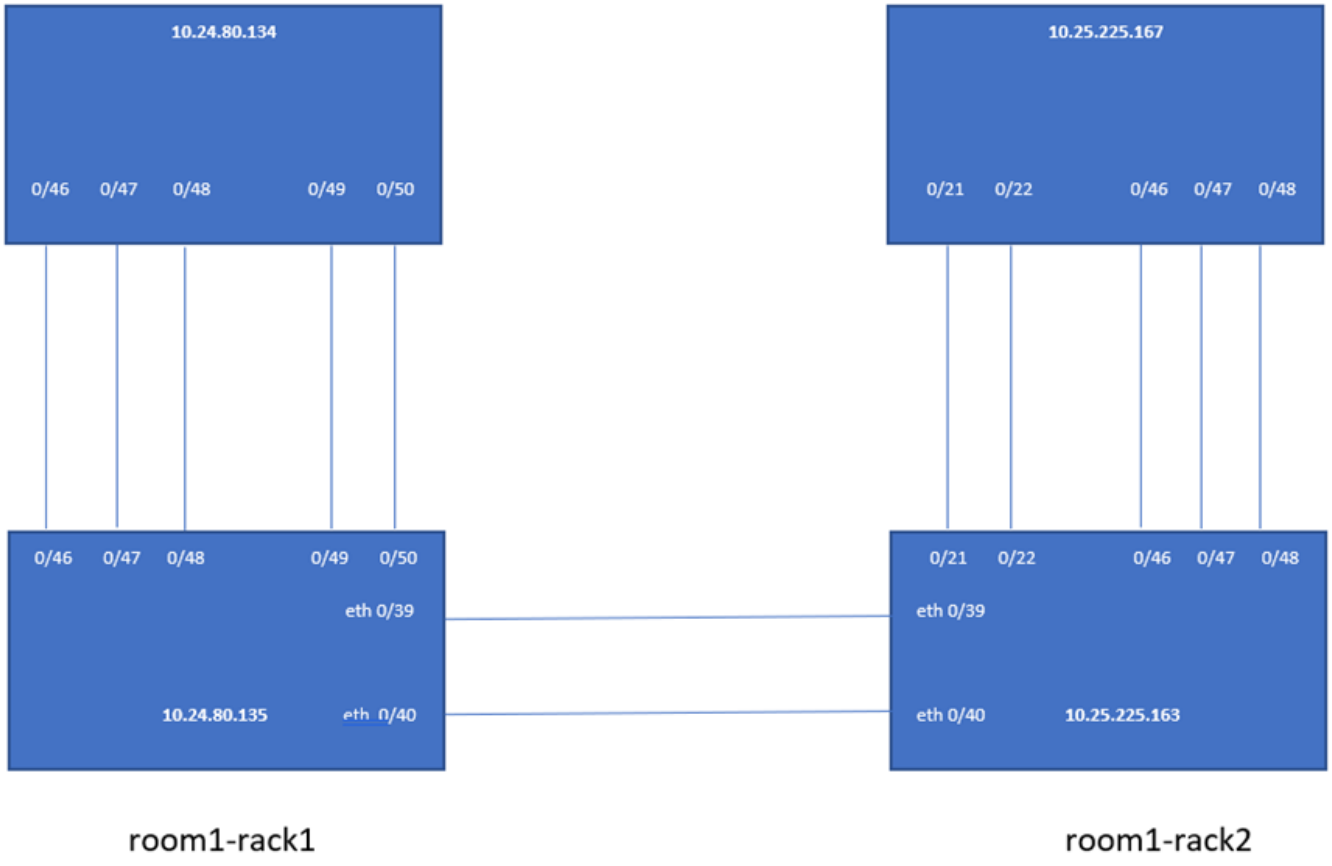


Figure 12: Multi-rack configuration example

Configure a Non-Clos Small Data Center Fabric

A non-Clos fabric is used in small data centers.

1. Create the fabric.

```
efa fabric create
```

2. Add a device to the fabric.

```
efa fabric device add
```

A device must be registered with Inventory Service before you can add it to a fabric. However, if you provide a user name and password when you run the command, then the devices are automatically registered with the Inventory Service. See the examples at the end of this procedure.

You can add multiple devices by using the `efa fabric device add-bulk` command. If you choose to add multiple devices in bulk, ensure you perform the following operations first:

- Run the **`efa inventory device register --ip <list-of-device-ips>`** command.
- Run the **`efa inventory device interface list --ip <device-ip>`** command. In the output of the command, verify that the states of the port links are as you expected (in the Admin Status and Oper Status fields). If not, manually check the physical cabling and fix any issues. Then continue with the **`efa fabric device add-bulk`** operation.

3. Configure the fabric.

```
$ efa fabric configure
```

Tip

The validation process reports any errors as a response to the **`efa fabric device add`** or **`efa fabric configure`** operations. You can use the **`efa fabric error show`** command to export these errors to a CSV file.

This example creates the fabric.

```
$ efa fabric create --name extr-fabric  --type non-clos
```

This example adds a device to the fabric.

```
efa fabric device add --name extr-fabric --ip 10.x.x.x --rack room1-rack1
--username admin --password password
```

This example adds multiple devices to the fabric.

```
$ efa fabric device add-bulk --name extr-fabric --rack room1-rack1
--ip 10.24.80.134,10.24.80.135 --rack room1-rack2 --ip 10.25.225.163,10.25.225.167
```

This example configures the fabric.

```
efa fabric configure --name extr-fabric
```

IP Multicast Fabric Provisioning

IP Multicast Fabric Overview

When multicast traffic is sent over unicast tunnels, ingress replication is done for each remote VTEP node. IP multicast fabric enables IP fabric to distribute BUM (Broadcast, Unknown Unicast, and Multicast Overlay) traffic using multicast VxLAN tunnels established over underlay fabric links.

Multicast Vxlan tunnels use Protocol Independent Multicast - Source Specific Multicast (PIM-SSM) and Multicast Distribution Tree (MDT) to deliver traffic effectively while minimizing packet replication in the fabric.

When multicast fabric is configured, a default MDT is created using PIM-SSM protocol running on fabric links and all the EVPN domain (VLANs/BDs) traffic is routed using the default tree.

The following figures show Clos topology for VxLAN unicast and multicast tunnels.

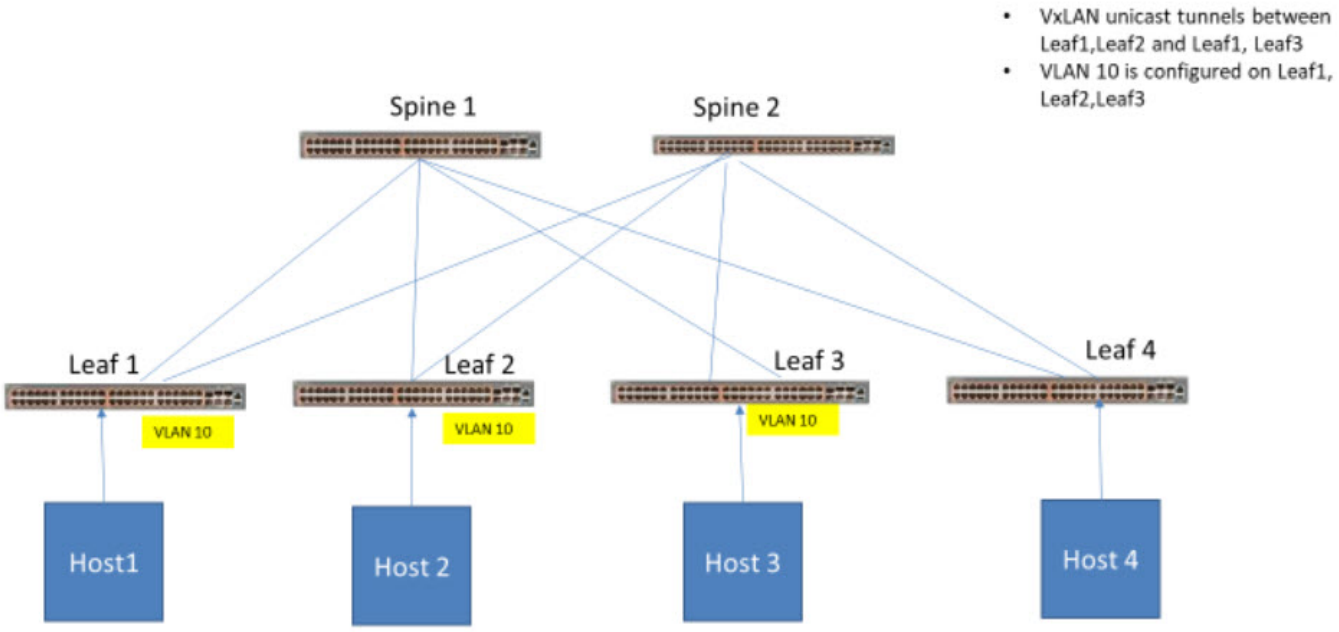


Figure 13: Clos topology with VxLAN unicast tunnels

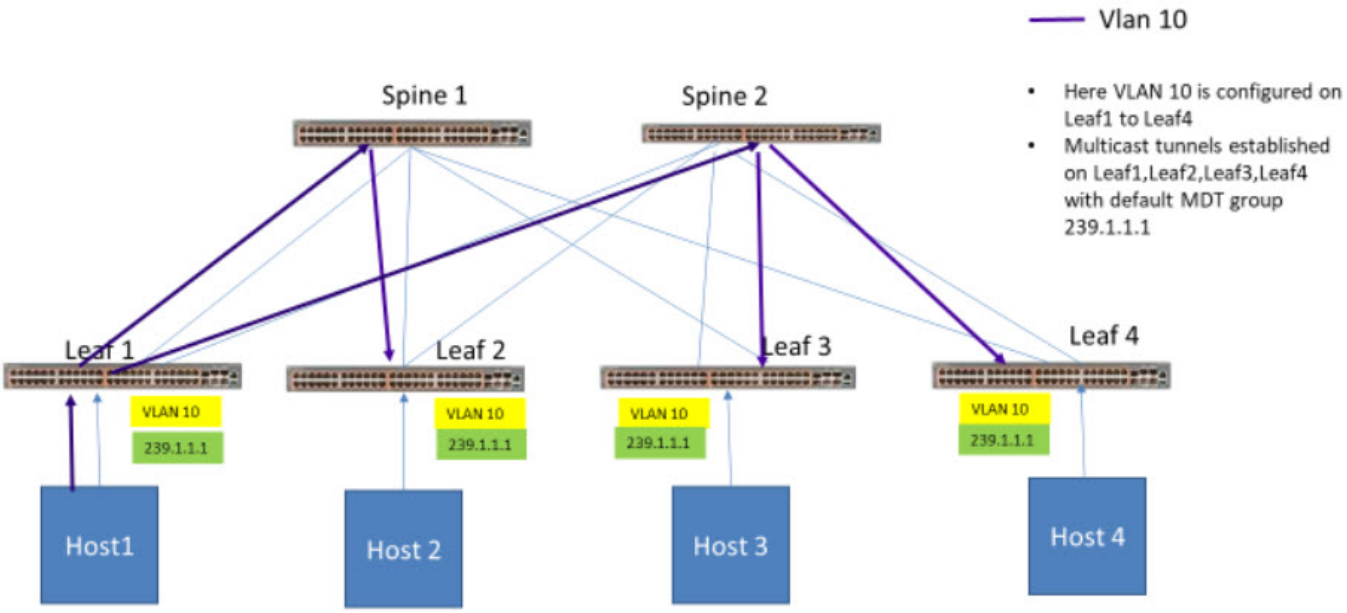


Figure 14: Clos topology with multicast tunnels

Supported Platforms

EFA supports IP multicast fabric configuration on both 3 stage and 5 stage Clos fabric on the following platforms:

- SLX 9540
- SLX 9640
- SLX 9150-48Y
- SLX 9150-48XT
- SLX 9250-32C
- SLX 9740

Note

- IP multicast fabric is not supported on non-Clos fabric.
- IPv6 multicast is not supported.

Bidirectional Forwarding Detection

Bidirectional Forwarding Detection (BFD) protocol detects faults between two forwarding engines.

When fabric is created, BFD is enabled by default along with fabric links and BGP neighbors. The following example shows BFD configuration settings.

```
# efa fabric setting show --name clos_fabric --advanced

+--------------------------------+-------------------------+
|              NAME              |          VALUE          |
+--------------------------------+-------------------------+
| Fabric Name                    | default                 |
```

```
+--------------------------------+------------------------+
| Link IP Range                  | 10.10.10.0/23          |
+--------------------------------+------------------------+
| Loopback IP Range              | 172.31.254.0/24        |
+--------------------------------+------------------------+
| Loopback Port Number           | 1                      |
+--------------------------------+------------------------+
| VTEP Loopback Port Number      | 2                      |
+--------------------------------+------------------------+
| Spine ASN Block                | 64512-64768            |
+--------------------------------+------------------------+
| SuperSpine ASN Block           | 64769                  |
+--------------------------------+------------------------+
| Leaf ASN Block                 | 65000-65534            |
+--------------------------------+------------------------+
| Border Leaf ASN Block          | 66000-66100            |
+--------------------------------+------------------------+
| P2P IP Type                    | numbered               |
+--------------------------------+------------------------+
| Any cast MAC                   | 0201.0101.0101         |
+--------------------------------+------------------------+
| IPV6 Any cast MAC              | 0201.0101.0102         |
+--------------------------------+------------------------+
| MAC Aging Timeout              | 1800                   |
+--------------------------------+------------------------+
| MAC Aging Conversational       | 300                    |
| Timeout                        |                        |
+--------------------------------+------------------------+
| MAC Move Limit                 | 20                     |
+--------------------------------+------------------------+
| Duplicate MAC Timer            | 5                      |
+--------------------------------+------------------------+
| Duplicate MAC Timer MAX Count  | 3                      |
+--------------------------------+------------------------+
| BFD Enable                     | Yes                    |
+--------------------------------+------------------------+
| BFD Tx                         | 300                    |
+--------------------------------+------------------------+
| BFD Rx                         | 300                    |
+--------------------------------+------------------------+
| BFD Multiplier                 | 3                      |
+--------------------------------+------------------------+
| BGP MultiHop                   | 2                      |
+--------------------------------+------------------------+
| MaxPaths                       | 8                      |
+--------------------------------+------------------------+
| AllowAsIn                      | 0                      |
+--------------------------------+------------------------+
| MTU                            | 9216                   |
+--------------------------------+------------------------+
| IPMTU                          | 9100                   |
+--------------------------------+------------------------+
| MCT Link IP Range              | 10.20.20.0/24          |
+--------------------------------+------------------------+
| MCT PortChannel                | 64                     |
+--------------------------------+------------------------+
| LACP Timeout                   | long                   |
+--------------------------------+------------------------+
| Control Vlan                   | 4090                   |
+--------------------------------+------------------------+
| Control VE                     | 4090                   |

…Skipped
```

Configure EFA IP Multicast Fabric

1. Create a Clos fabric.

```
# efa fabric create --name clos_fabric --type clos --stage 3
```

Note
Optimized replication is not supported on non-Clos fabric.

2. Enable multicast fabric settings.

```
# efa fabric setting update --optimized-replication-enable yes --name clos_fabric
```

3. (Optional) Override the default MDT group and group range.

```
# efa fabric setting update --name clos_fabric --mdtgroup-range <A.B.C.D/L> --default-
mdtgroup <A.B.C.D>

# efa fabric setting update --name clos_fabric --mdtgroup-range 239.0.0.0/8 --default-
mdtgroup 239.1.1.1
```

4. Verify the fabric settings.

```
# efa fabric setting show --name clos_fabric --advanced

+--------------------------------+------------------------+
| Optimized Replication Enable   | Yes                    |
+--------------------------------+------------------------+
| MDT Group IPv4 Range           | 238.0.0.0/8            |
+--------------------------------+------------------------+
| Default MDT Group IPv4 address | 238.1.1.1              |
+--------------------------------+------------------------+
```

5. Add devices to the fabric.

```
# efa fabric device add-bulk --name clos_fabric --leaf Leaf1IP,Leaf2IP,Leaf3IP,Leaf4IP
--spine Spine1IP,Spine2IP --username admin --password password
```

6. Configure the fabric.

```
# efa fabric configure -name clos_fabric
```

7. Verify fabric configuration.

```
# efa fabric show-config --name clos_fabric --advanced
```

- All underlay configuration and overlay configurations are pushed to the devices and underlay topology is operational.
- All BGP connections between leaf and spine nodes are established and neighbors are reachable.
- Basic overlay configuration with optimized replication is configured.
- All device configurations are applied to the devices in fabric. For more information, see Device Configuration on page 67.

8. Create a tenant and EPG to bring up the multicast tunnels with leaf nodes.

```
# efa tenant create --name tenant1 --l2-vni-range 10002-14190 --l3-vni-range
14191-14200 --vrf-count 10 --vlan-range 2-4090 --port
Leaf1IP[0/12-16],Leaf2IP[0/12-16], Leaf3IP[0/12-16],Leaf4IP[0/12-16] --description
Subscriber1

# efa tenant epg create --name epg1 --tenant tenant1 --port
Leaf1IP[0/15],Leaf2IP[0/15],Leaf3IP[0/16 --switchport-mode trunk --ctag-range 100
```

Device Configuration

When IP multicast fabric is enabled, the following device configurations are pushed to all devices in the fabric.

- `router pim` for default VRF is enabled on all nodes.
- `ip prefix-list` is configured on all nodes with the `mdt-range` specified in fabric settings or default range.
- Under `router-pim` mode, PIM-SSM is enabled for all nodes with range specified in `ip prefix-list`.
- Under `interface` mode, `PIM sparse` mode is enabled for all fabric links.
- Under overlay-gateway, `optimized replication` is enabled on all leaf nodes.
- Under `optimized replication` mode, `underlay-default-mdtgroup` is configured to default value specified in fabric settings on all leaf nodes.

Configure Drift and Reconcile on Multicast Fabric

1. Configure drift and reconcile on multicast fabric.

```
# efa fabric debug device drift --ip A.B.C.D --name dni --reconcile
```

Note

Any drift in Router PIM, IP prefix-list, and overlay-gateway EFA configuration compared to the configured device is detected and reconciled.

2. Configure drift and reconcile of all services on a device.

```
# efa inventory drift-reconcile execute --ip A.B.C.D --reconcile
```

Viewing Fabric Details

You can use several commands to view the details of topologies and configuration in your fabric.

Table 9: Fabric show commands

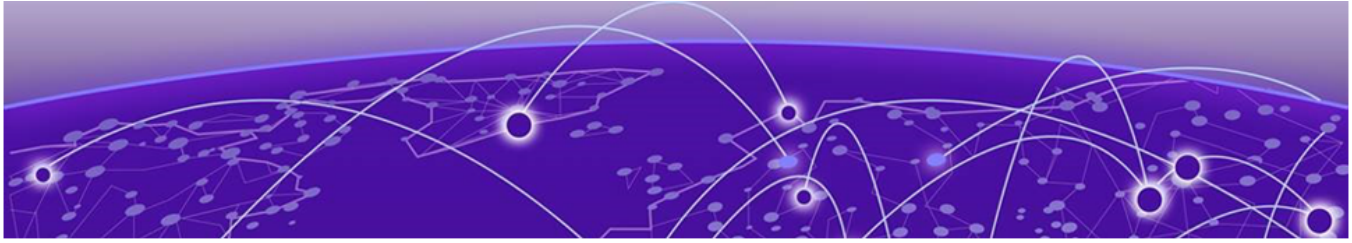
Command	Description
efa fabric topology show overlay	Shows the overlay (VxLAN tunnels) connectivity of devices in a fabric.
efa fabric topology show physical	Shows the physical topology (LLDP neighbors) connectivity of devices in a fabric.
efa fabric topology show underlay	Shows the underlay (BGP neighbors) connectivity of devices in a fabric.
efa fabric show	Displays fabric details.
efa fabric show-config	Displays fabric configuration details for the specified role (leaf, spine, super-spine, border leaf) or IP address.
efa fabric show summary	Displays a summary of all fabrics or of the specified fabric.

The following is an example of output from the **efa fabric topology show overlay** command.

```
efa fabric topology show overlay --name fabric1
+------------+--------------------------+--------------------------+----------------
+---------------+-------------+------------+----------
| OVERLAY    |      SOURCE LEAF IP      |    DESTINATION LEAF IP   | SOURCE VTEP IP |
DESTINATION   | OVERLAY     | OVERLAY    | OVERLAY
  ECAP TYPE
VTEP IP         ADMIN STATE   OPER STATE   BFD STATE
+------------+--------------------------+--------------------------+----------------
+---------------+-------------+------------+----------
| vxlan      | 10.25.225.11,10.25.225.46 | 10.24.85.76,10.24.85.74   | 172.31.254.86  |
172.31.254.81 | up          | up         | down
| vxlan      | 10.25.225.11,10.25.225.46 | 10.24.80.134,10.24.80.135 | 172.31.254.86  |
172.31.254.83 | up          | up         | down
| vxlan      | 10.24.85.76,10.24.85.74   | 10.25.225.11,10.25.225.46 | 172.31.254.81  |
172.31.254.86 | up          | up         | down
| vxlan      | 10.24.85.76,10.24.85.74   | 10.24.80.134,10.24.80.135 | 172.31.254.81  |
172.31.254.83 | up          | up         | down
| vxlan      | 10.24.80.134,10.24.80.135 | 10.25.225.11,10.25.225.46 | 172.31.254.83  |
172.31.254.86 | up          | up         | down
| vxlan      | 10.24.80.134,10.24.80.135 | 10.24.85.76,10.24.85.74   | 172.31.254.83  |
172.31.254.81 | up          | up         | down
+------------+--------------------------+--------------------------+----------------
+---------------+-------------+------------+----------
```

The following is an example of output from the **efa fabric show** command.

```
efa fabric topology show overlay --name fabric1
+--------------+-------+-------------+-----------+------+--------------+-------------
+------------+----------+------+-------
| IP ADDRESS   | RACK  |  HOST NAME  | ASN       | ROLE | DEVICE STATE | APP STATE   |
CONFIG     | PENDING  | VTLB | LB ID

GEN REASON   CONFIGS    ID
+--------------+-------+-------------+-----------+------+--------------+-------------
+------------+----------+------+-------
| 10.24.51.131 | rack2 | Freedom-07  | 4200000001| leaf | provisioned  | cfg in-sync |
NA         | NA       | 2    | 1
| 10.25.255.58 | rack2 | Freedom-04  | 4200000001| leaf | provisioned  | cfg in-sync |
NA         | NA       | 2    | 1
| 10.24.51.135 | rack1 | Freedom-06  | 4200000000| leaf | provisioned  | cfg in-sync |
NA         | NA       | 2    | 1
| 10.24.48.131 | rack1 | Freedom-05  | 4200000002| leaf | provisioned  | cfg in-sync |
NA         | NA       | 2    | 1
```



Tenant Services Provisioning

Tenant Services Provisioning Overview

Tenant Services exposes the CLI and REST API for automating the Tenant network configuration on the Clos and Non-Clos overlay fabric.

Tenant network configuration includes VLAN, BD, VE, EVPN, VTEP, VRF, and Router BGP configuration on the necessary fabric devices to provide L2-Extension, L3-Extension across the fabric, L2-Handoff, and L3-Handoff at the edge of the fabric.

Tenant Services provisioning automates the Tenant configuration, which can be a subset of the combinations provided by the switching hardware.

Tenant Services supports multiple fabrics.

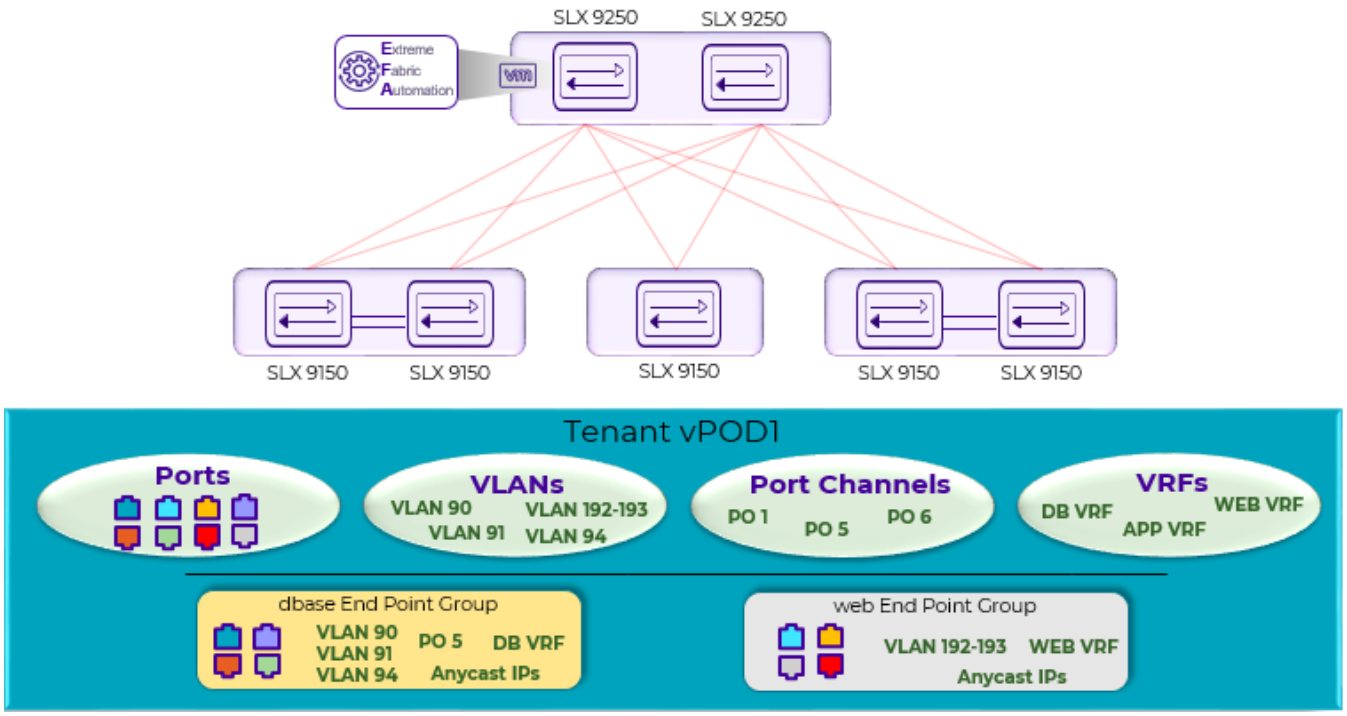


Figure 15: Tenant Services Overview

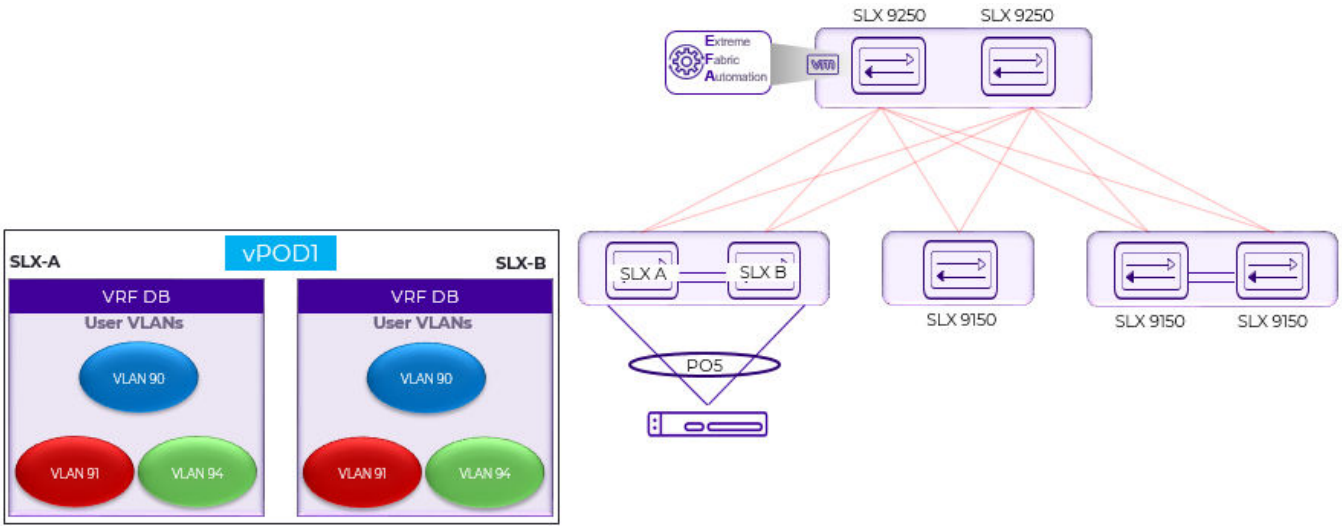


Figure 16: Tenant Name vPOD1, VRF Name DB

Tenant

A Tenant is a logical construct that owns resources as follows:

- VLAN range: Ctags pertaining to which the traffic is expected to ingress and egress.

  Note
  Ctag (Customer VLAN tag) is used to identify the customer broadcast domain. In the IP Fabric network, it represents the customer and is mapped into a VXLAN tunnel thru a VNI (virtual network identifier). The VNI is the ID used to identify the VXLAN tunnel. With auto VNI mapping the Ctag ID equals the VNI. Users can also manually map Ctags to user-defined VNIs. These VNIs can be VLAN IDs (up to 4k) or to BD's (bridge domain) IDs.

- Device ports: Ports on which the traffic is expected to ingress and egress.

VLAN-based Tenant

For a VLAN based tenant, realization of network on the device is done using VLAN and switchport VLANs. Bridge domains are used for EVPN IRB.

Bridge domain-based Tenant

For a BD based tenant, realization of network on the device is done using BD and BD-LIF. BD is used for EVPN IRB.

Scalability

Table 10: VNI scalability

VNI type	Scale
Non-auto VNI mapping	• The number of VNI (networks) supported per device = 8K [4K VLAN + 4K BD] • The maximum number of VNI (networks) supported in the fabric = [8K * number of devices in the fabric].
Auto VNI mapping	• The number of VNI (networks) supported per device = 8K [4K VLAN + 4K BD] • The number of VNI (networks) supported per fabric = 8K

Event handling

Event handling specifies the scope of the tenant configuration on the devices.

Devices are added to the Tenant service only when the Fabric is provisioned on the devices.

An event is an occurrence of a device being removed from the Fabric or from the Inventory.

- When a device is removed from the Fabric or Inventory, the device is cleaned up from Tenant Service and the Tenant configuration is removed from the device.

- User-created entities, such as Tenant, VRF, and EPG, are not deleted whereas references for ports/port-channels of deleted devices are removed.

Provisioning a Tenant

This high-level process describes the tasks that you complete to provision a tenant in your EFA fabric.

A tenant is a group of users that own or have access to shared resources.

Create a tenant

This step in the process provides a name for the tenant and identifies the resources that are reserved for the tenant, including the Layer 2 and Layer 3 Virtual Network Identifiers (VNI), VLAN, VRF, and bridge domain. You can later apply these resources to an endpoint group.

Use the **`efa tenant create`** command to create your tenant. For syntax and command examples, see the Extreme Fabric Automation Command Reference, 2.4.0.

For related information, see the following topics.

- Clos Fabric with Auto VNI Map on page 75
- VRF: Maximum-Paths on page 91
- Sharing Resources Across Tenants

Create a port channel

This step in the process creates the port channel for the tenant. A port channel, also known as a Link Aggregation Group (LAG) is a communication link between devices. You can specify speed, LACP negotiation, port, port channel number, LACP timeout, and the number of links that are required to be up.

Use the **`efa tenant po create`** command to create the port channel. For syntax and command examples, see the *Extreme Fabric Automation Command Reference*.

For related information, see the following topics.

- EPG: CEP Cluster Tracking on page 103
- Exclusion of VLANs and Bridge from Cluster Instance on page 112
- Sharing Resources Across Tenants
- Port Channel: Minimum Link Count on page 81

Create the tenant VRF

This step in the process sets up virtual routing and forwarding (VRF) for the tenant. You can specify the VRF name and the associated tenant, the target VPN community, the Route Target and Route Distinguisher, the local ASN, IPv4 and IPv6 static BFD routes, IPv4 and IPv6 static next hop routes, the number of load sharing paths, the redistribute type, whether resilient hashing is on SLX devices, and the routing type.

Use the **efa tenant vrf create** command to configure VRF. For syntax and command examples, see the *Extreme Fabric Automation Command Reference*.

For related information, see the following topics.

- IPv6 Support on page 112
- VRF: Static Route on page 85
- VRF: BFD on Static Route on page 87
- VRF: Local-ASN on page 88
- VRF: Resilient Hashing on page 90
- VRF: Maximum-Paths on page 91
- VRF: Redistribute on page 93
- Sharing Resources Across Tenants

Create the tenant endpoint group

This step in the process creates the endpoint group for the tenant. An endpoint group is a logical group of endpoints, which are devices that are connected to the network. You can specify such parameters as group name, the IP address, the port channels, the switchport mode, the BGP service type, native VLAN, CTAG range, the associated VRF, the Layer 2 and Layer 3 VNI, the bridge domain, and neighbor discovery preferences.

Use the **efa tenant epg create** command to create the endpoint group. For syntax and command examples, see the *Extreme Fabric Automation Command Reference*.

For related information, see the following topics.

- Clos Fabric with Auto VNI Map on page 75
- IPv6 Support on page 112
- VRF: Static Route on page 85
- VRF: BFD on Static Route on page 87
- EPG: CEP Cluster Tracking on page 103
- Exclusion of VLANs and Bridge from Cluster Instance on page 112
- VRF: Maximum-Paths on page 91
- VRF: Redistribute on page 93

Create the BGP peer group

This step in the process creates the Border Gateway Protocol (BGP) peer group, which is a set of BGP neighbors that share outbound policies. You can specify the group name, the group ASN, the BFD (bidirectional forwarding detection) properties of the group, source IP and next hop information, and the name of the associated tenant.

Use the **efa tenant service bgp peer-group create** command to create the peer group. For syntax and command examples, see the *Extreme Fabric Automation Command Reference*.

For related information, see BGP Peer Group on page 106.

Create the BGP peer

This step in the process creates the BGP peer for the tenant. BGP peers are devices that exchange BGP routing information. You can specify the IPv4 and IPv6 dynamic unicast neighbors, the IPv4 and IPv6 unicast neighbors, and the IPv4 and IPv6 BFD unicast neighbors. Additional parameters include next-hop-self information, source IP address, the name of the peer, and the name of the associated tenant.

Use the **`efa tenant service bgp peer create`** command to create the peer. For syntax and command examples, see the *Extreme Fabric Automation Command Reference*.

For related information, see BGP Peer on page 108.

Clos Fabric with Non-auto VNI Maps

Auto VNI simplifies the mapping IDs by using the VLAN ID as the VNI ID, for example VLAN 100 = VNI 100.

This method of mapping works well in environments where overlapping VLANs are not being used. However, if two different tenants are using VLAN 100, VNI 100 cannot be used by both. At this point, manual mapping of VLAN to VNI is required. Extreme Fabric Automation simplifies this process by allowing VNI ranges for tenants to automate “manual” mapping to work for overlapping VLANs.

The following figure shows a 3-stage Clos topology.

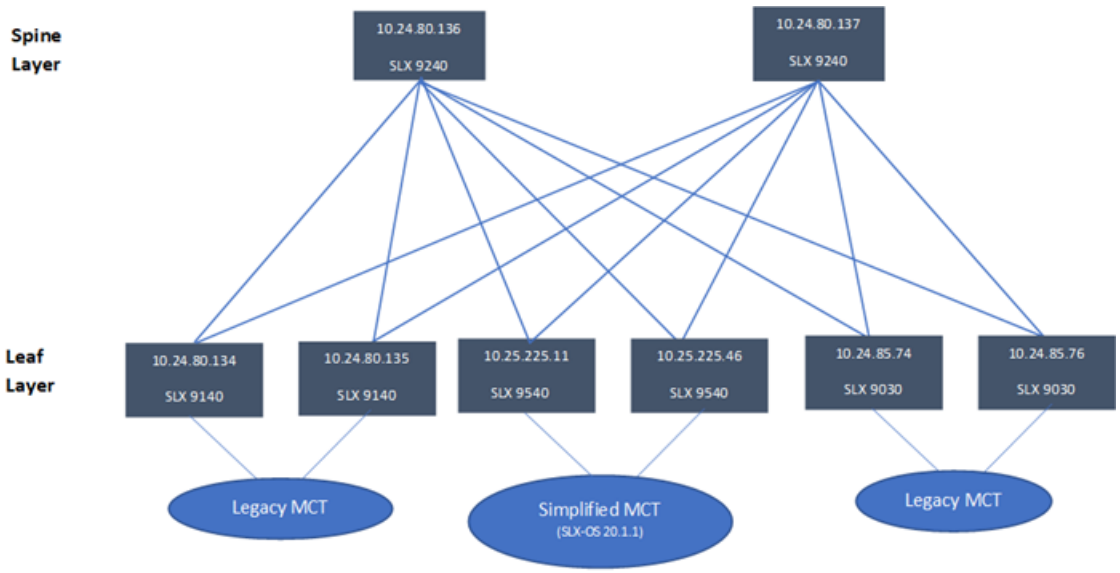


Figure 17: 3-stage Clos topology

The following commands configure the 3-stage Clos topology:

```
efa fabric create --name fabric1

efa fabric setting update --name fabric1 --vni-auto-map No

efa fabric device add-bulk --spine 10.24.80.136 --border-leaf 10.25.225.11,10.25.225.46
--leaf 10.24.80.134-135,10.24.85.74,10.24.85.76 --username admin --password password
--name fabric1

efa fabric configure --name fabric1
```

The following figure shows tenant constructs in the Clos Fabric.

Figure 18: Scope of tenant constructs

Clos Fabric with Auto VNI Map

- In Clos fabric with auto VNI map, the VNI is statically derived using the VLAN ID or BD ID.
  - For the VLAN case, VNI = VLAN ID
  - For the BD case, VNI = 4096 + BD ID
  - User will not be able to reserve l2-vni-range or l3-vni-range for a given tenant.
  - User will not be able to provide a specific l2-vni/l3-vni in an EPG.
- VLAN Based Tenants:

  Multiple VLAN Based tenants cannot share the same VLAN, considering the multiple tenants cannot share the same VNI.
- BD Based Tenants:

  Multiple BD Based tenants can share the same VLAN, as the VLANs from each tenant will be mapped to a unique BD and further a unique VNI.

Multi Tenancy

EFA supports multi tenancy by allowing multiple tenants to have overlapping ctags and non-overlapping L2VNI. A tenant ctag will get a unique L2VNI and a unique network allocated in the fabric

The following example shows a multi tenancy configuration.

```
efa tenant create --name tenant11 --vrf-count 10 --vlan-range 2-4090 --port
10.24.80.134[0/15-17],10.24.80.135[0/15-17],10.25.225.11[0/15-17],10.25.225.46[0/15-17],10
.24.85.74[0/15-17],10.24.85.76[0/15-17] --description Subscriber1

efa tenant show
+----------+-------------+-------------+------------+-----------+-----------
+-----------------------+
|   Name   | L2VNI-Range | L3VNI-Range | VLAN-Range | VRF-Count | Enable-BD |
Ports         |
+----------+-------------+-------------+------------+-----------+-----------
+-----------------------+
| tenant11 |             |             | 2-4090     | 10        | False     |
10.24.85.74[0/15-17]  |
|          |             |             |            |           |           |
10.24.80.135[0/15-17] |
|          |             |             |            |           |           |
10.25.225.11[0/15-17] |
|          |             |             |            |           |           |
10.25.225.46[0/15-17] |
```

```
|          |             |             |            |           |           |
10.24.80.134[0/15-17] |
|          |             |             |            |           |           |
10.24.85.76[0/15-17]  |
+----------+-------------+-------------+------------+-----------+-----------
+-----------------------+

efa tenant create --name tenant12 --vrf-count 10  --vlan-range 2-4090 --port
10.24.80.134[0/18-20],10.24.80.135[0/18-20],10.25.225.11[0/18-20],10.25.225.46[0/18-20],10
.24.85.74[0/18-20],10.24.85.76[0/18-20]
Tenant Creation Failed:
         Vlan (2) overlaps with Tenant (tenant11)

efa tenant create --name tenant21 --vrf-count 10 --enable-bd --port
10.24.80.134[0/21-25],10.24.80.135[0/21-25],10.24.85.74[0/21-25],10.24.85.76[0/21-25],10.2
5.225.11[0/21-25],10.25.225.46[0/21-25]

efa tenant create --name tenant22 --vrf-count 10 --enable-bd --port
10.24.80.134[0/26-30],10.24.80.135[0/26-30],10.24.85.74[0/26-30],10.24.85.76[0/26-30],10.2
5.225.11[0/26-30],10.25.225.46[0/26-30]

efa tenant show
+----------+-------------+-------------+------------+-----------+-----------
+-----------------------+
|   Name   | L2VNI-Range | L3VNI-Range | VLAN-Range | VRF-Count | Enable-BD |
Ports         |
+----------+-------------+-------------+------------+-----------+-----------
+-----------------------+
| tenant11 |             |             | 2-4090     | 10        | False     |
10.25.225.46[0/15-17] |
|          |             |             |            |           |           |
10.25.225.11[0/15-17] |
|          |             |             |            |           |           |
10.24.80.135[0/15-17] |
|          |             |             |            |           |           |
10.24.85.74[0/15-17]  |
|          |             |             |            |           |           |
10.24.85.76[0/15-17]  |
|          |             |             |            |           |           |
10.24.80.134[0/15-17] |
+----------+-------------+-------------+------------+-----------+-----------
+-----------------------+
| tenant21 |             |             | 2-4090     | 10        | True      |
10.24.85.74[0/21-25]  |
|          |             |             |            |           |           |
10.25.225.11[0/21-25] |
|          |             |             |            |           |           |
10.25.225.46[0/21-25] |
|          |             |             |            |           |           |
10.24.80.134[0/21-25] |
|          |             |             |            |           |           |
10.24.85.76[0/21-25]  |
|          |             |             |            |           |           |
10.24.80.135[0/21-25] |
+----------+-------------+-------------+------------+-----------+-----------
+-----------------------+
| tenant22 |             |             | 2-4090     | 10        | True      |
10.24.85.76[0/26-30]  |
|          |             |             |            |           |           |
10.25.225.11[0/26-30] |
|          |             |             |            |           |           |
10.24.80.135[0/26-30] |
|          |             |             |            |           |           |
10.25.225.46[0/26-30] |
```

```
|          |             |             |            |           |           |
10.24.85.74[0/26-30]  |
|          |             |             |            |           |           |
10.24.80.134[0/26-30] |
+----------+-------------+-------------+------------+-----------+-----------
+------------------------+

efa tenant epg create --name epg11 --tenant tenant11 --po po1115,po1215,po1315 --
switchport-mode trunk --switchport-native-vlan 11 --ctag-range 11-12

efa tenant epg create --name epg21 --tenant tenant21 --po po2121,po2221,po2321 --
switchport-mode trunk  --ctag-range 11-12

efa tenant epg create --name epg22 --tenant tenant21 --po po2122,po2222,po2322  --
switchport-mode trunk  --ctag-range 11-12

efa tenant epg show

=========================================================================
Name          : epg11
Tenant        : tenant11
Description   :
Ports         :
POs           : po1315, po1215, po1115
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 11-12

Network Property [Flags : * - Native Vlan]
+------+--------+------------+---------+-------------+-------------+
| Ctag | L2-Vni | Anycast-ip | BD-name |  Dev-state  |  App-state  |
+------+--------+------------+---------+-------------+-------------+
| 11*  | 11     |            |         | provisioned | cfg-in-sync |
+------+--------+------------+---------+-------------+-------------+
| 12   | 12     |            |         | provisioned | cfg-in-sync |
+------+--------+------------+---------+-------------+-------------+

=========================================================================
=========================================================================
Name          : epg21
Tenant        : tenant21
Description   :
Ports         :
POs           : po2121, po2221, po2321
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 11-12

Network Property [Flags : * - Native Vlan]
+------+--------+------------+--------------+-------------+-------------+
| Ctag | L2-Vni | Anycast-ip |   BD-name    |  Dev-state  |   App-state |
+------+--------+------------+--------------+-------------+-------------+
| 11   | 4099   |            | Auto-BD-4099 | provisioned | cfg-in-sync |
+------+--------+------------+--------------+-------------+-------------+
| 12   | 4100   |            | Auto-BD-4100 | provisioned | cfg-in-sync |
+------+--------+------------+--------------+-------------+-------------+

=========================================================================
=========================================================================
Name          : epg22
Tenant        : tenant21
Description   :
Ports         :
POs           : po2122, po2222, po2322
```

```
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 11-12

Network Property [Flags : * - Native Vlan]
+------+--------+------------+--------------+-------------+-------------+
| Ctag | L2-Vni | Anycast-ip |   BD-name    |  Dev-state  |  App-state  |
+------+--------+------------+--------------+-------------+-------------+
| 12   | 4102   |            | Auto-BD-4102 | provisioned | cfg-in-sync |
+------+--------+------------+--------------+-------------+-------------+
| 11   | 4101   |            | Auto-BD-4101 | provisioned | cfg-in-sync |
+------+--------+------------+--------------+-------------+-------------+

efa tenant epg create --name epg23 --tenant tenant21 --po po2122,po2322  --switchport-
mode trunk  --ctag-range 21-22 --bridge-domain 21:Auto-BD-4101 --bridge-domain 22:Auto-
BD-4102

efa tenant epg show

=========================================================================
Name          : epg11
Tenant        : tenant11
Description   :
Ports         :
POs           : po1315, po1215, po1115
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 11-12

Network Property [Flags : * - Native Vlan]
+------+--------+------------+---------+-------------+-------------+
| Ctag | L2-Vni | Anycast-ip | BD-name |  Dev-state  |  App-state  |
+------+--------+------------+---------+-------------+-------------+
| 11*  | 11     |            |         | provisioned | cfg-in-sync |
+------+--------+------------+---------+-------------+-------------+
| 12   | 12     |            |         | provisioned | cfg-in-sync |
+------+--------+------------+---------+-------------+-------------+

=========================================================================
=========================================================================
Name          : epg21
Tenant        : tenant21
Description   :
Ports         :
POs           : po2121, po2221, po2321
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 11-12

Network Property [Flags : * - Native Vlan]
+------+--------+------------+--------------+-------------+-------------+
| Ctag | L2-Vni | Anycast-ip |   BD-name    |  Dev-state  |   App-state |
+------+--------+------------+--------------+-------------+-------------+
| 11   | 4099   |            | Auto-BD-4099 | provisioned | cfg-in-sync |
+------+--------+------------+--------------+-------------+-------------+
| 12   | 4100   |            | Auto-BD-4100 | provisioned | cfg-in-sync |
+------+--------+------------+--------------+-------------+-------------+

=========================================================================
=========================================================================
Name          : epg22
Tenant        : tenant21
Description   :
Ports         :
```

```
POs           : po2122, po2222, po2322
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 11-12

Network Property [Flags : * - Native Vlan]
+------+--------+------------+--------------+-------------+-------------+
| Ctag | L2-Vni | Anycast-ip |   BD-name    |  Dev-state  |  App-state  |
+------+--------+------------+--------------+-------------+-------------+
| 12   | 4102   |            | Auto-BD-4102 | provisioned | cfg-in-sync |
+------+--------+------------+--------------+-------------+-------------+
| 11   | 4101   |            | Auto-BD-4101 | provisioned | cfg-in-sync |
+------+--------+------------+--------------+-------------+-------------+

=========================================================================
=========================================================================
Name          : epg23
Tenant        : tenant21
Description   :
Ports         :
POs           :
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 21-22

Network Property [Flags : * - Native Vlan]
+------+--------+------------+--------------+-------------+-------------+
| Ctag | L2-Vni | Anycast-ip |   BD-name    |  Dev-state  |  App-state  |
+------+--------+------------+--------------+-------------+-------------+
| 21   | 4101   |            | Auto-BD-4101 | provisioned | cfg-in-sync |
+------+--------+------------+--------------+-------------+-------------+
| 22   | 4102   |            | Auto-BD-4102 | provisioned | cfg-in-sync |
+------+--------+------------+--------------+-------------+-------------+

=========================================================================
```

Layer 3 Network Services

The topics in this section describe Tenant provisioning in the Layer 3 network.

Port Channel: Description

The feature let you know the “description” per EFA port-channel which further gets configured on the SLX port-channel. The default value of PO “description” is **“EFA Port-channel <efa-po-name>”.**

You can provide the PO “description” during PO create and PO update operations.

2.3.2 to 2.4.0 Upgrade Handling

All the POs upgraded from EFA 2.3.2 to EFA 2.4.0 version must have the default “description” and is displayed as part of **`efa tenant po show --detail`**.

Port-Channel Create

```
efa tenant po create --name <po-name> --tenant <tenant-name> --description <po-
description>
               --speed <100Mbps|1Gbps|10Gbps|25Gbps|40Gbps|100Gbps> --negotiation <active|
passive|static>
```

```
    --port <list-of-po-members> --min-link-count <min-link-count> --number <po-number> --
lacp-timeout <short|long>
```

Port-Channel Update

```
efa tenant po update --name <po-name> --tenant <tenant-name>
              --operation <port-add|port-delete|lacp-timeout|description|min-link-count>
              --port <list-of-po-members> --lacp-timeout string <short|long> --min-link-
count <min-link-count>
              --description <po-description>
```

Example

```
efa tenant po create --name ten1po1 --tenant ten1 --port
10.20.246.15[0/1],10.20.246.16[0/1] --speed 10Gbps --negotiation active --description
tenant1po1


efa tenant po create --name ten1po2 --tenant ten1 --port
10.20.246.15[0/2],10.20.246.16[0/2] --speed 10Gbps --negotiation active

efa tenant po update --name ten1po1 --tenant ten1 --operation description --description
tenat1po1changed
```

efa tenant po show --name ten1po1 --tenant ten1 --detail	**efa tenant po show --name ten1po2 --tenant ten1 --detail**
Name : ten1po1 Tenant : ten1 ID : 1 Description : tenat1po1changed Speed : 10Gbps Negotiation : active Min Link Count : 1 Lacp Timeout : long Ports : 10.20.246.15[0/1] : 10.20.246.16[0/1] State : po-created Dev State : provisioned App State : cfg-in-sync	Name : ten1po2 Tenant : ten1 ID : 2 Description : EFA Port-channel ten1po2 Speed : 10Gbps Negotiation : active Min Link Count : 1 Lacp Timeout : long Ports : 10.20.246.15[0/2] : 10.20.246.16[0/2] State : po-created Dev State : provisioned App State : cfg-in-sync

Switch Config

```
Rack1-Device1# show running-
config interface Port-channel
interface Port-channel 1
 description tenat1po1changed
 no shutdown
!
interface Port-channel 2
 description EFA Port-channel ten1po2
 no shutdown
!
Rack1-Device1#
```

```
Rack1-Device2# show running-
config interface Port-channel
interface Port-channel 1
 description tenat1po1changed
 no shutdown
!
interface Port-channel 2
 description EFA Port-channel ten1po2
 no shutdown
!
Rack1-Device2#
```

Port Channel: Minimum Link Count

Minimum number of links per port-channel indicates the least number of links that need to be operationally UP in order to declare the port-channel as operationally UP. Users can provide an optional "min-link-count" per EFA port-channel during the PO create and PO update operations, which further gets configured on the SLX port-channel, .

Default value of min-link-count for a port-channel is 1, which is same as the SLX default value.

Update the min-link-count attribute value using `min-link-count` operation.

Provide the PO “min-link-count” during PO create and PO update operations. EFA validates the po-member-port-count >= po-min-link-count per device

Note

During the upgrade from EFA 2.3.0 to EFA 2.4.0, the min-link-count for the port-channels is set to the default value 1.

2.3.2 to 2.4.0 Upgrade Handling

- Empty PO: EFA 2.4.0 does not support empty PO, hence during the upgrade from EFA 2.3.2 to EFA 2.4.0, all the empty POs are marked with “delete-pending” state.
- Non empty PO: During the upgrade from EFA 2.3.2 to EFA 2.4.0, all the non-empty PO get configured with the default value (1) of “min-link-count”, and this value is displayed as part of “efa tenant po show” output.

Note

Single Homed to Dual Homed PO conversion is not allowed in EFA 2.4.0.

Port-Channel Create

```
efa tenant po create --name <po-name> --tenant <tenant-name> --description <po-
description>
        --speed <100Mbps|1Gbps|10Gbps|25Gbps|40Gbps|100Gbps> --negotiation <active|
passive|static>
     --port <list-of-po-members> --min-link-count <min-link-count>
        --number <po-number> --lacp-timeout <short|long>
```

Port-Channel Update

```
efa tenant po update --name <po-name> --tenant <tenant-name>
         --operation <port-add|port-delete|lacp-timeout|description|min-link-count>
         --port <list-of-po-members> --lacp-timeout string <short|long> --min-link-count
<min-link-count>
         --description <po-description>
```

Example

```
efa tenant po create --name ten1po1 --tenant ten1 --port
10.20.246.15[0/1-2],10.20.246.16[0/1-2]
--speed 10Gbps --negotiation active --description tenant1po1 --min-link-count 2

efa tenant po create --name ten1po2 --tenant ten1 --port
10.20.246.15[0/3],10.20.246.16[0/3]
--speed 10Gbps --negotiation active
```

```
efa tenant po update --name ten1po1 --tenant ten1 --operation port-delete --port
10.20.246.15[0/1],10.20.246.16[0/1] --min-link-count 1

efa tenant po update --name ten1po1 --tenant ten1 --operation port-add --port
10.20.246.15[0/1],10.20.246.16[0/1] --min-link-count 2

efa tenant po update --name ten1po1 --tenant ten1 --operation min-link-count --min-link-
count 1

efa tenant po update --name ten1po1 --tenant ten1 --operation min-link-count --min-link-
count 2

efa tenant po show
+---------+--------+----+--------+-------------+----------+---------+--------------------
+------------+-------------+-------------+
|  Name   | Tenant | ID | Speed  | Negotiation | Min Link |  Lacp   |        Ports
|   State    |  Dev State  |  App State  |
|         |        |    |        |             |  Count   | Timeout |
|           |             |             |
+---------+--------+----+--------+-------------+----------+---------+--------------------
+------------+-------------+-------------+
| ten1po1 |  ten1  | 1  | 10Gbps |   active    |    2     |  long   | 10.20.246.15[0/1-2]
| po-created | provisioned | cfg-in-sync |
|         |        |    |        |             |          |         | 10.20.246.16[0/1-2]
|           |             |             |
+---------+--------+----+--------+-------------+----------+---------+--------------------
+------------+-------------+-------------+
| ten1po2 |  ten1  | 2  | 10Gbps |   active    |    1     |  long   |  10.20.246.15[0/3]
| po-created | provisioned | cfg-in-sync |
|         |        |    |        |             |          |         |  10.20.246.16[0/3]
|           |             |             |
+---------+--------+----+--------+-------------+----------+---------+--------------------
+------------+-------------+-------------+
```

Switch Config

```
Rack1-Device1# show running-
config interface Port-channel
interface Port-channel 1
 minimum-links 2
 description tenant1po1
 no shutdown
!
interface Port-channel 2
 description EFA Port-channel ten1po2
 no shutdown
!
Rack1-Device1#
```

```
Rack1-Device2# show running-
config interface Port-channel
interface Port-channel 1
 minimum-links 2
 description tenant1po1
 no shutdown
!
interface Port-channel 2
 description EFA Port-channel ten1po2
 no shutdown
!
Rack1-Device2#
```

VRF: Backup Routing

Backup routing needs to be enabled when all links from a leaf device to the spine layer are down and tenant traffic is to be routed via the MCT neighbor.

EFA Provisioning

IPv4 and IPv6 range will be input at the fabric level. An IPv4 and an IPv6 address pair will be allocated to every MCT pair across all the tenant VRFs and the BGP session will be established between the same IP Pair.

Allocations happen per device:

1. Allocate a Bridge domain per VRF for backup routing.
2. Allocate a corresponding router-interface VE per BD/VRF.
3. Assign the IPv4 and IPv6 address allocated to each device on each of the VE interface.

Note
Same IPv4 and IPv6 address will be allocated on each of the VE interface belonging to different VRF.

4. Establish IBGP IPV4 neighborship with the MCT peer on a set of IP address per VRF.
5. Establish IBGP IPv6 neighborship with the MCT peer on a set of IPv6 address per VRF.
6. Configure "next-hop-self" on both the IPv4 and IPv6 neighbor.
7. Configure "active" on the IPv6 neighbor.

Example:

```
efa fabric setting update --name fabric1
--backup-routing-ipv4-range 21.1.1.0/24 --backup-routing-ipv6-range 2001:21:1:1::0/120
```

Example where backup routing is enabled:

```
efa fabric setting update --name nc --backup-routing-enable yes
```

Device Config

Table 11: Tenant1 VRF "vrf1"

leaf-9250-173# show running-config bridge-domain 3001 bridge-domain 3001 p2mp pw-profile default router-interface Ve 7001 bpdu-drop-enable ! leaf-9250-173# sh run in ve 7001 interface Ve 7001 vrf forwarding vrf1 ip address 21.1.1.10/31 ipv6 address 2001:21:1:1::10/127 no shutdown !	leaf-9250-173# show running-config router bgp address-family ipv4 unicast vrf vrf1 router bgp address-family ipv4 unicast vrf vrf1 local-as 4210000001 redistribute connected neighbor 21.1.1.11 remote-as 4210000001 neighbor 21.1.1.11 next-hop-self maximum-paths 2 ! ! leaf-9250-173# show running-config router bgp address-family ipv6 unicast vrf vrf1 router bgp address-family ipv6 unicast vrf vrf1 redistribute connected neighbor 2001:21:1:1::11 next-hop-self neighbor 2001:21:1:1::11 remote-as 4210000001 neighbor 2001:21:1:1::11 activate maximum-paths 2 ! !

Table 12: Tenant2 VRF "vrf2"

leaf-9250-173# show running-config bridge-domain 3002 bridge-domain 3002 p2mp pw-profile default router-interface Ve 7002 bpdu-drop-enable ! leaf-9250-173# sh run in ve 7002 interface Ve 7002 vrf forwarding vrf2 ip address 21.1.1.10/31 ipv6 address 2001:21:1:1::10/127 no shutdown !	leaf-9250-173# show running-config router bgp address-family ipv4 unicast vrf vrf2 router bgp address-family ipv4 unicast vrf vrf2 local-as 4210000001 redistribute connected neighbor 21.1.1.11 remote-as 4210000001 neighbor 21.1.1.11 next-hop-self maximum-paths 2 ! ! leaf-9250-173# show running-config router bgp address-family ipv6 unicast vrf vrf2 router bgp address-family ipv6 unicast vrf vrf2

Table 12: Tenant2 VRF “vrf2” (continued)

	redistribute connected neighbor 2001:21:1:1::11 next-hop-self neighbor 2001:21:1:1::11 remote-as 4210000001 neighbor 2001:21:1:1::11 activate maximum-paths 2 ! !

VRF: Static Route

The Static Route Configuration at the tenant VRF level allows you to provide static routes per tenant VRF.

VRF Create

```
# efa tenant vrf create [ --name vrf name | --tenant tenant name | --rttype <both |
import | export > | --rt | --ipv4-static-route-next-hop < device ip,ipv4 static route
network,nexthop ip,nexthop distance > | --ipv6-static-route-next-hop < device ip,ipv4
static route network,nexthop ip,nexthop distance > | --local-asn | --ipv4-static-route-
bfd < device-ip,dest-ipv4-addr,source-ipv4-addr[interval,minrx,multiplier] > | --ipv6-
static-route-bfd < device-ip,dest-ipv4-addr,source-ipv4-addr[interval,min-rx,multiplier]
> | --max-path uint <1-64> | --redistribute < static | connected >| --rh-max-path uint |
--rh-ecmp-enable | --help ]
```

VRF Update

```
# efa tenant vrf update [ --name <vrf-name> --tenant <tenant-name> --operation <
staticroute-add | static-route-delete> --ipv6static-route-next-hop <destination, next-
hop> --ipv4static-route-next-hop <destination, next-hop> efa tenant vrf update [ --name
vrf name | --tenant tenant name | --operation < local-asn-add | local-asn-delete | static-
route-bfd-add | static-route-bfd-delete | static-route-add | static-route-delete | max-
path-add | max-path-delete | redistribute-add | redistributedelete | rh-max-path-add | rh-
max-path-delete | rh-ecmp-update > | --local-asn | --ipv4-static-route-bfd < device
IP,dest-ipv4-addr,source-ipv4-addr[interval,min-rx,multiplielr] > | --ipv6-staticroute-
bfd < device ip,dest-ipv6-addr,source-ipv6-addr[interval,minrx,multiplielr] > | --ipv4-
static-route-next-hop < device-ip,destipv4-addr,source-ipv4-addr[interval,min-
rx,multiplier] > | --ipv6-static-route-next-hop < device-ip,dest-ipv4-addr,source-ipv4-
addr[interval,min-rx,multiplier] > | --max-path uint <1-64> | --redistribute < static |
connected > | --rh-max-path uint | --rh-ecmpenable | --help ]
```

```
# efa tenant vrf create --name red --tenant tenant11 --ipv6-static-route-next-hop
10.24.80.134,2000::/64,1001::2 --ipv6-static-route-next-hop
10.24.80.134,2000::/64,1002::2 --ipv6-static-route-next-hop
10.24.80.134,2000::/64,1003::2 --ipv6-static-route-next-hop
10.24.80.134,2000::/64,1004::2 --ipv6-static-route-next-hop
10.24.80.135,2001::/64,1001::2,4 --ipv6-static-route-next-hop
10.24.80.135,2001::/64,1002::2 --ipv6-static-route-next-hop
10.24.80.135,2001::/64,1003::2 --ipv6-static-route-next-hop
10.24.80.135,2001::/64,1004::2 --ipv4-static-route-next-hop
10.24.80.134,22.0.0.0/24,13.0.0.1,2 --ipv4-static-route-next-hop
10.24.80.134,22.0.0.0/24,13.0.0.2 --ipv4-static-route-next-hop
10.24.80.134,22.0.0.0/24,13.0.0.3 --ipv4-static-route-next-hop
10.24.80.134,22.0.0.0/24,13.0.0.4 --ipv4-static-route-next-hop
10.24.80.135,23.0.0.0/24,13.0.0.1 --ipv4-static-route-next-hop
10.24.80.135,23.0.0.0/24,13.0.0.2 --ipv4-static-route-next-hop
```

```
10.24.80.135,23.0.0.0/24,13.0.0.3 --ipv4-static-route-next-hop
10.24.80.135,23.0.0.0/24,13.0.0.4

# efa tenant vrf show --name red --tenant tenant11

================================================================================Name
          : red
Tenant Name     : tenant11
L3 VNI          :
Route Target    :
Static Route    : Switch-IP, Network->Nh1[distance],Nh2[distance],..
                : 10.24.80.134, 22.0.0.0/24->13.0.0.1[2],13.0.0.2,13.0.0.3,13.0.0.4
                  10.24.80.135, 23.0.0.0/24->13.0.0.1,13.0.0.2,13.0.0.3,13.0.0.4
                  10.24.80.134, 2000::/64->1001::2,1002::2,1003::2,1004::2
                  10.24.80.135, 2001::/64->1001::2[4],1002::2,1003::2,1004::2
Local Asn       :
================================================================================

efa tenant epg create --name ten1epg1 --tenant tenant1 --port
10.24.80.134[0/11],10.24.80.135[0/11] --switchport-mode trunk –ctag-range 11 --vrf red –
anycast-ip 11:10.10.11.1/24
```

Switch Config

```
Device 1# show running-config vrfred
vrf red
address-family ipv4 unicast
  route-target export 1:1 evpn
  route-target import 1:1 evpn
 ip route 22.0.0.0/24 13.0.0.1 distance 2
  ip route 22.0.0.0/24 13.0.0.2
  ip route 22.0.0.0/24 13.0.0.3
  ip route 22.0.0.0/24 13.0.0.4
  !
 address-family ipv6 unicast
  route-target export 1:1 evpn
  route-target import 1:1 evpn
  ipv6 route 2000::/64 1001::2
  ipv6 route 2000::/64 1002::2
  ipv6 route 2000::/64 1003::2
  ipv6 route 2000::/64 1004::2
 !


Device 2# show running-config vrfred
vrf red
address-family ipv4 unicast
  route-target export 1:1 evpn
  route-target import 1:1 evpn
 ip route 22.0.0.0/24 13.0.0.1
  ip route 22.0.0.0/24 13.0.0.2
  ip route 22.0.0.0/24 13.0.0.3
  ip route 22.0.0.0/24 13.0.0.4
  !
 address-family ipv6 unicast
  route-target export 1:1 evpn
  route-target import 1:1 evpn
  ipv6 route 2000::/64 1001::2 distance 4
  ipv6 route 2000::/64 1002::2
  ipv6 route 2000::/64 1003::2
  ipv6 route 2000::/64 1004::2
 !
```

VRF: BFD on Static Route

Provides an option at the tenant VRF level so static route BFD timers can be used.

VRF Create

```
# efa tenant vrf create --name <vrf-name> --tenant <tenant-name>
--ipv6static-route-bfd <destination-ip, source-ip, bfd-min-tx, bfd-min-rx, bfd-
multiplier>
--ipv4static-route-bfd < destination-ip, source-ip, bfd-min-tx, bfd-min-rx, bfd-
multiplier>
```

VRF Update

```
# efa tenant vrf update -name <vrf-name> --tenant <tenant-name> --operation <static-route-
bfd-add|static-route-bfd-delete> --ipv6-static-route-bfd <switch-ip, destination-ip,
source-ip, bfd-min-tx, bfd-min-rx, bfd-multiplier> --ipv4static-route-bfd  <switch-ip,
destination-ip, source-ip, bfd-min-tx, bfd-min-rx, bfd-multiplier>
```

```
# efa tenant vrf create --name red --tenant tenant11 --ipv6-static-route-bfd
10.24.80.134,1001::2,1001::1,100,200,5 --ipv6-static-route-bfd
10.24.80.135,1011::2,1011::1,100,200,5 --ipv6-static-route-bfd 10.24.80.134,1002::2,
1002::1 --ipv6-static-route-bfd 10.24.80.135,1012::2, 1012::1 --ipv4-static-route-bfd
10.24.80.134,13.0.0.1,13.0.0.9,200,300,6 --ipv4-static-route-bfd
10.24.80.135,13.0.1.1,13.0.1.9,200,300,6 --ipv4-static-route-bfd
10.24.80.134,13.0.0.2,13.0.0.10 --ipv4-static-route-bfd 10.24.80.135,13.0.1.2,13.0.1.10

# efa tenant epg create --name ten1epg1 --tenant tenant1 --port
10.24.80.134[0/11],10.24.80.135[0/11] --switchport-mode trunk -ctag-range 11 --vrf red -
anycast-ip 11:10.10.11.1/24
```

Switch Config

```
vrf vrf1
 rd 172.31.254.13:1
 evpn irb ve 8192
 address-family ipv4 unicast
  route-target export 1:1 evpn
  route-target import 1:1 evpn
  ip route 22.0.0.0/24 13.0.0.1
  ip route 22.0.0.0/24 13.0.0.2
  ip route 22.0.0.0/24 13.0.0.3
  ip route 22.0.0.0/24 13.0.0.4
  ip route 22.0.0.0/24 13.0.0.5
  ip route static bfd 13.0.0.1 13.0.0.9 interval 200 min-rx 300 multiplier 6
  ip route static bfd 13.0.0.2 13.0.0.10
 !
 address-family ipv6 unicast
  route-target export 1:1 evpn
  route-target import 1:1 evpn
  ipv6 route 2000::/64 1001::2
  ipv6 route 2000::/64 1002::2
  ipv6 route 2000::/64 1003::2
  ipv6 route 2000::/64 1004::2
  ipv6 route 2000::/64 1005::2
  ipv6 route static bfd 1001::2 1001::1 interval 100 min-rx 200 multiplier 5
  ipv6 route static bfd 1002::2 1002::1
 !
```

VRF: Local-ASN

At the tenant VRF level, this option allows for providing local-asn per tenant VRF.

EFA Provisioning

```
efa tenant vrf create --name <vrf-name> --tenant <tenant-name> --local-asn <local-as-for-vrf>
efa tenant vrf update --name <vrf-name> --tenant <tenant-name> --operation <local-asn-add|local-asn-delete> --local-asn <value>
```

Note
The local-asn support on IPv6 AF must be checked.

Switch Config

leaf-9250-173# sh run router bgp router bgp **local-as 4200000000** capability as4-enable fast-external-fallover bfd interval 100 min-rx 100 multiplier 3 neighbor 2001:11:1::2 remote-as 4200000001 neighbor 2001:11:1::2 bfd neighbor 2001:12:1::2 remote-as 4200000001 neighbor 2001:12:1::2 bfd neighbor 2001:91:1::1 remote-as 4230000000 neighbor 10.20.20.10 remote-as 4200000000 neighbor 10.20.20.10 next-hop-self neighbor 10.20.20.10 bfd neighbor 11.1.0.2 remote-as 4200000001 neighbor 11.1.0.2 bfd neighbor 12.1.0.2 remote-as 4200000001 neighbor 12.1.0.2 bfd neighbor 91.1.0.1 remote-as 4230000000 !	address-family ipv4 unicast vrf vrf1 **local-as 4210000001** redistribute connected neighbor 11.1.1.2 remote-as 4220000001 neighbor 11.1.1.2 bfd neighbor 12.1.1.2 remote-as 4220000001 neighbor 12.1.1.2 bfd neighbor 21.1.1.1 remote-as 4210000001 neighbor 21.1.1.1 next-hop-self neighbor 91.1.1.1 remote-as 4230000001 maximum-paths 2 ! address-family ipv4 unicast vrf vrf10 **local-as 4210000010** redistribute connected neighbor 11.1.10.2 remote-as 4220000010 neighbor 11.1.10.2 bfd neighbor 12.1.10.2 remote-as 4220000010 neighbor 12.1.10.2 bfd neighbor 21.1.10.1 remote-as 4210000010 neighbor 21.1.10.1 next-hop-self neighbor 91.1.10.1 remote-as 4230000010 maximum-paths 2 !
leaf-9250-173# show running-config router bgp address-family ipv4 unicast vrf vrf1 router bgp address-family ipv4 unicast vrf vrf1 local-as 4210000001 maximum-paths 2 !	leaf-9250-173# show running-config router bgp address-family ipv6 unicast vrf vrf1 router bgp **address-family ipv6 unicast vrf vrf1** **redistribute connected** **maximum-paths 2** !

VRF: Graceful Restart

The Tenant service configures graceful restart under each tenant VRF ipv4/v6 unicast address-family.

You can enable graceful restart per tenant VRF during VRF creation and VRF update operations. EFA provisions the user input graceful restart on the switches during the instantiation and update of VRF on the switches.

- VRF is instantiated on the switches during a user-triggered L3 EPG create or L2 EPG transition to L3 EPG, based on the endpoints present in the EPG.
- VRF is updated on the switches during a user-triggered VRF update operation, based on the endpoints present in the EPGs, using this VRF.

Currently, the enable or disable value of graceful restart depends on the per-user VRF configuration.

Note

- By default, graceful restart is disabled. The default value of graceful restart is the switch default.
- EFA versions earlier than v2.4 have graceful restart enabled by default.
- An upgrade from a version earlier than v2.4 to v2.4 will retain the graceful restart as enabled for all the VRFs that were configured before the upgrade.

EFA Provisioning

- When using the **`efa tenant vrf create`** command, add the **`--graceful-restart`** parameter. For more information, see the **`efa tenant vrf create`** command in the Extreme Fabric Automation Command Reference, 2.4.0.
- When using the **`efa tenant vrf update`** command, add the **`graceful-restart-update`** option for the **`--operation`** parameter and **`--graceful-restart`** parameter. For more information, see the **`efa tenant vrf update`** command in the Extreme Fabric Automation Command Reference, 2.4.0.

Switch Config

Rack1-Device1#sh run router bgp router bgp local-as 4200000000! address-family ipv4/v6 unicast vrf vrf1 redistribute connected graceful-restart ! address-family ipv4 unicast vrf vrf10 redistribute connected !	Rack1-Device2#sh run router bgp router bgp local-as 4200000000! address-family ipv4/v6 unicast vrf vrf1 redistribute connected graceful-restart ! address-family ipv4 unicast vrf vrf10 redistribute connected !

VRF: Resilient Hashing

As a load-balancing method, resilient hashing helps to lessen the possibility that a destination path will be remapped when a LAG (Link Aggregation Group) link fails.

When you create or update a tenant VRF, you can enable ECMP resilient hashing and you can configure the maximum number of resilient hashing paths allowed (8, 16, or 64 paths). Resilient hashing is disabled by default. The default number of allowed paths is the same as the default value for the SLX devices.

When you create a tenant VRF, use the **efa tenant vrf create** command with the **--rh-ecmp-enable** and **--rh-max-path** parameters to enable resilient hashing and set the maximum number of allowed paths. For example:

```
# efa tenant vrf create --name <vrf-name> --tenant <tenant-name>
--rh-ecmp-enable=true/false --rh-max-path <8 | 16 | 64>
```

When you update a tenant VRF, use the **efa tenant vrf update** command with the **--rh-ecmp-enable** and **--rh-max-path** parameters to enable resilient hashing and set the maximum number of allowed paths. For example:

```
# efa tenant vrf update --name <vrf-name> --tenant <tenant-name>
--operation <rh-max-path-add | rh-max-path-delete | rh-ecmp-update>
--rh-ecmp-enable=true/false --rh-max-path <8 | 16 | 64>
```

The **--max-path** and **--rh-max-path** parameters can co-exist.

You cannot choose the specific devices on which to configure resilient hashing. Configuration applies to all SLX devices in the tenant VRF.

For more information about the commands, including usage examples, see the Extreme Fabric Automation Command Reference, 2.4.0.

VRF: Graceful Restart (GR)

User can enable GR (graceful-restart) per tenant VRF during VRF create and VRF update operations. EFA provisions the user input GR on the switches during the instantiation and updation of VRF on the switches.

VRF is instantiated on the switches during the user triggered L3 EPG create or L2 EPG transition to L3 EPG, based on the endpoints present in the EPG.

VRF is updated on the switches during the user triggered VRF update operation, based on the endpoints present in the EPGs using this VRF.

Currently GR gets configured wherever the VRF gets configured.

Note

1. By default, the GR is disabled. Default value of GR is the switch default.
2. Pre-EFA 2.4.0 versions have GR enabled by default.
3. Upgrade from pre-2.4.0 to 2.4.0 retains the GR enabled for all the pre-2.4.0 VRFs.

VRF Create

```
efa tenant vrf create --name <vrf-name> --tenant <tenant-name>
            --graceful-restart=true/false
```

VRF Update

```
efa tenant vrf update --name <vrf-name> --tenant <tenant-name>
            --operation graceful-restart-update --graceful-restart=true/false
```

Example

```
efa tenant vrf create --name vrf1 --tenant tenant1
efa tenant vrf create -name vrf10 -tenant tenant1 --graceful-restart=true

efa tenant epg create --name ten1epg1 --tenant tenant1
            --port 10.24.80.134[0/11],10.24.80.135[0/11]
            --switchport-mode trunk -ctag-range 11 --vrf vrf1 -anycast-ip 11:10.10.11.1/24
efa tenant epg create --name ten1epg2 --tenant tenant1
            --port 10.24.80.134[0/12],10.24.80.135[0/12]
            --switchport-mode trunk -ctag-range 12 --vrf vrf10 -anycast-ip 12:10.10.12.1/24
efa tenant vrf update --name vrf1 --tenant tenant1
            --operation graceful-restart-update --graceful-restart=true
efa tenant vrf update --name vrf10 --tenant tenant1
            --operation graceful-restart-update -graceful-restart=false
```

Switch Config

```
Rack1-Device1# sh run router bgp
router bgp
 local-as 4200000000!
address-family ipv4/v6 unicast vrf vrf1
  redistribute connected
  graceful-restart
 !
address-family ipv4 unicast vrf vrf10
  redistribute connected
 !
```

```
Rack1-Device2# sh run router bgp
router bgp
 local-as 4200000000!
address-family ipv4/v6 unicast vrf vrf1
  redistribute connected
    graceful-restart
 !
address-family ipv4 unicast vrf vrf10
  redistribute connected
 !
```

VRF: Maximum-Paths

EFA allows provisioning of maximum-paths per tenant VRF during VRF create and VRF update operations.

VRF is updated on the switches during the user triggered VRF update operation based on the endpoints present in the EPGs using the VRF.

Note

Default value of `max-path` is 8.

Choosing specific devices for `max-path` provisioning is not allowed.

EFA Provisioning

```
# efa tenant vrf create --name <vrf-name> --tenant <tenant-name> --max-path <value>

# efa tenant vrf update --name <vrf-name> --tenant <tenant-name> --operation <max-path-
add|max-path-delete> --max-path <value>
```

```
# efa tenant vrf create --name vrf1 --tenant tenant1

# efa tenant vrf create -name vrf10 -tenant tenant1

# efa tenant epg create --name ten1epg1 --tenant tenant1 --
port10.24.80.134[0/11],10.24.80.135[0/11] --switchport-mode trunk -ctag-range 11 --vrf
vrf1 --anycast-ip 11:10.10.11.1/24

# efa tenant epg create --name ten1epg2 --tenant tenant1 --
port10.24.80.134[0/12],10.24.80.135[0/12] --switchport-mode trunk -ctag-range 12 --vrf
vrf10 --anycast-ip 12:10.10.12.1/24

# efa tenant vrf update --name vrf10 -tenant tenant1 --operation max-paths-add --max-path
13 Switch Config
```

Switch Config

```
Device1# sh run router bgp
router bgp
local-as 4200000000!
address-family ipv4unicast vrf vrf1
redistribute connected
maximum-paths8
!
address-family ipv4unicast vrf vrf10
redistribute connected
maximum-paths13
!
address-family ipv6unicast vrf vrf1
redistribute connected
maximum-paths8
!
address-family ipv6unicast vrf vrf10
redistribute connected
maximum-paths13
!
Device2# sh run router bgp
router bgp
local-as 4200000000!
address-family ipv4unicast vrf vrf1
redistribute connected
maximum-paths8
!
address-family ipv4unicast vrf vrf10
redistribute connected
maximum-paths13
!
address-family ipv6unicast vrf vrf1
redistribute connected
maximum-paths8
!
address-family ipv6unicast vrf vrf10
redistribute connected
maximum-paths13
!
```

VRF: Redistribute

EFA allows provisioning of redistribute attribute per tenant VRF during VRF create and VRF update operations.

VRF is updated on the switches during the user triggered VRF update operation based on the endpoints present in the EPGs using the VRF.

Note

Default value of `redistribute` is `connected`.

Choosing specific devices for `redistribute` provisioning is not allowed.

EFA provisioning

VRF Create

```
# efa tenant vrf create --name <vrf-name> --tenant <tenant-name> --redistribute <list>
```

VRF Update

```
# efa tenant vrf update --name <vrf-name> --tenant <tenant-name> --operation
<redistribute-add|redistribute-delete> --redistribute <static | connected>
```

VRF Update

```
# efa tenant vrf create --name vrf1 --tenant tenant1 --redistribute static

# efa tenant vrf create -name vrf10 -tenant tenant1

# efa tenant epg create --name ten1epg1 --tenant tenant1 --
port10.24.80.134[0/11],10.24.80.135[0/11] --switchport-mode trunk -ctag-range 11 --vrf
vrf1 --anycast-ip 11:10.10.11.1/24

# efa tenant epg create --name ten1epg2 --tenant tenant1 --
port10.24.80.134[0/12],10.24.80.135[0/12] --switchport-mode trunk --ctag-range 12 --vrf
vrf10 --anycast-ip 12:10.10.12.1/24

# efa tenant vrf update --name vrf10 -tenant tenant1 --operation redistribute-add --
redistribute static
Device1# sh run router bgp
router bgp
local-as 4200000000
address-family ipv4unicast vrf vrf1
redistribute static
!
address-family ipv4unicast vrf vrf10
redistribute connected
redistribute static
!
Device2# sh run router bgp
router bgp
local-as 4200000000
address-family ipv4unicast vrf vrf1
redistribute static
!
address-family ipv4unicast vrf vrf10
redistribute connected
redistribute static
```

```
!
address-family ipv6unicast vrf vrf1
redistribute static
!
!address-family ipv6unicast vrf vrf10
redistribute connected
redistribute static
!
```

EPG: Network Property Description

The EPG Network Property Description allows you to configure “description” per EFA tenant ctag which gets configured on the SLX as VLAN/BD description.

The default value of “description” is as below:

Network Type	Description
L2 Extension	Tenant L2 Extended VLAN/BD
L3 Extension	Tenant L3 Extended VLAN/BD
L3 Handoff	Tenant L3 Hand-off VLAN/BD
L3 Extension EVPN IRB	Tenant L3 Extended IRB BD

You can provide the EPG network “description” during EPG create and EPG update (ctag-range-add) operations.

2.3.2 to 2.4.0 Upgrade Handling

During the upgrade from EFA 2.3.2 to EFA 2.4.0, for all the EPG networks, the default “description” is determined and the app-state is set to “cfg-refreshed” so that the user triggers DRC (in EFA 2.4.0) to push the derived network “description” on to the SLX.

EPG Create

```
efa tenant epg create --name <epg-name> --tenant string <tenant-name> --port <list-of-
phy> --po <list-of-po>
               --switchport-mode <access |trunk | trunk-no-default-native>
               --ctag-range string <ctag-range> --ctag-description <ctag:description>
```

EPG Update

```
efa tenant epg update --name <epg-name> --tenant <tenant-name> --operation <ctag-range-
add>
                --ctag-range <ctag-range> --ctag-description <ctag:description>
```

Example

```
efa tenant show
+----------+---------+------------+-------------+-------------+-----------+-----------
+-----------------------+
|   Name   |  Type   | VLAN Range | L2VNI Range | L3VNI Range | VRF Count | Enable BD
|         Ports         |
+----------+---------+------------+-------------+-------------+-----------+-----------
+-----------------------+
```

```
|  bdTen1  | private |   21-30    |             |             |    10     |   true    |
10.20.246.15[0/11-20] |
|          |         |            |             |             |           |           |
10.20.246.16[0/11-20] |
+----------+---------+------------+-------------+-------------+-----------+-----------
+------------------------+
| vlanTen1 | private |   11-20    |             |             |    10     |   false   |
10.20.246.16[0/1-10]  |
|          |         |            |             |             |           |           |
10.20.246.15[0/1-10]  |
+----------+---------+------------+-------------+-------------+-----------+-----------
+------------------------+
efa tenant vrf create -name ten1vrf1 -tenant vlanTen1

efa tenant vrf create -name ten2vrf2 -tenant bdTen1
```

VLAN Based L2 Extension EPG

```
efa tenant epg create --name ten1epg1 --tenant vlanTen1 --port
10.20.246.15[0/1],10.20.246.16[0/1]
--switchport-mode trunk --ctag-range 11-12 --ctag-description 12:Ten1VLANNW1
```

VLAN Based L3 Extension EPG:

```
efa tenant epg create --name ten1epg2 --tenant vlanTen1 --port
10.20.246.15[0/1],10.20.246.16[0/1]
--switchport-mode trunk --ctag-range 13-14 --ctag-description 14:Ten1VLANNW2 --anycast-ip
13:10.0.13.1/24 --anycast-ip 14:10.0.14.1/24 --vrf ten1vrf1
```

VLAN Based L3 Handoff EPG

```
efa tenant epg create --name ten1epg3 --tenant vlanTen1 --type l3-hand-off
--port 10.20.246.15[0/1],10.20.246.16[0/1] --switchport-mode trunk --ctag-range 15-16 --
ctag-description 16:Ten1VLANNW3
```

BD Based L2 Extension EPG

```
efa tenant epg create --name ten2epg1 --tenant bdTen1 --port
10.20.246.15[0/11],10.20.246.16[0/11]
--switchport-mode trunk --ctag-range 21-22 --ctag-description 22:Ten2BDNW1
```

BD Based L3 Extension EPG

```
efa tenant epg create --name ten2epg2 --tenant bdTen1 --port
10.20.246.15[0/11],10.20.246.16[0/11]
--switchport-mode trunk --ctag-range 23-24 --ctag-description 24:Ten2BDNW2 --anycast-ip
23:10.0.23.1/24 --anycast-ip 24:10.0.24.1/24 --vrf ten2vrf
```

efa tenant epg show --detail

Switch Config

```
Rack1-Device1# show running-
config bridge-domain
bridge-omain 1 p2mp
 description Tenant L2 Extended BD
 pw-profile default
 logical-interface ethernet 0/11.21
 bpdu-drop-enable
 local-switching
!
bridge-domain 2 p2mp
 description Ten2BDNW1
 pw-profile default
 logical-interface ethernet 0/11.22
 bpdu-drop-enable
 local-switching
!
bridge-domain 3 p2mp
 description Tenant L3 Extended BD
 pw-profile Tenant-profile
 router-interface Ve 4099
 !
 logical-interface ethernet 0/11.23
 bpdu-drop-enable
 local-switching
 suppress-arp
!
bridge-domain 4 p2mp
 description Ten2BDNW2
 pw-profile Tenant-profile
 router-interface Ve 4100
 !
 logical-interface ethernet 0/11.24
 bpdu-drop-enable
 local-switching
 suppress-arp
!
bridge-domain 4093 p2mp
 description Tenant L3 Extended IRB BD
 pw-profile Tenant-profile
 router-interface Ve 8189
 !
 bpdu-drop-enable
 local-switching
!
bridge-domain 4094 p2mp
 description Tenant L3 Extended IRB BD
 pw-profile Tenant-profile
 router-interface Ve 8190
 !
 bpdu-drop-enable
 local-switching
!
```

```
Rack1-Device2# show running-
config bridge-domain
bridge-domain 1 p2mp
 description Tenant L2 Extended BD
 pw-profile default
 logical-interface ethernet 0/11.21
 bpdu-drop-enable
 local-switching
!
bridge-domain 2 p2mp
 description Ten2BDNW1
 pw-profile default
 logical-interface ethernet 0/11.22
 bpdu-drop-enable
 local-switching
!
bridge-domain 3 p2mp
 description Tenant L3 Extended BD
 pw-profile Tenant-profile
 router-interface Ve 4099
 !
 logical-interface ethernet 0/11.23
 bpdu-drop-enable
 local-switching
 suppress-arp
!
bridge-domain 4 p2mp
 description Ten2BDNW2
 pw-profile Tenant-profile
 router-interface Ve 4100
 !
 logical-interface ethernet 0/11.24
 bpdu-drop-enable
 local-switching
 suppress-arp
!
bridge-domain 4093 p2mp
 description Tenant L3 Extended IRB BD
 pw-profile Tenant-profile
 router-interface Ve 8189
 !
 bpdu-drop-enable
 local-switching
!
bridge-domain 4094 p2mp
 description Tenant L3 Extended IRB BD
 pw-profile Tenant-profile
 router-interface Ve 8190
 !
 bpdu-drop-enable
 local-switching
!
```

```
Rack1-Device1#show running-config vlan
vlan 11
 description Tenant L2 Extended VLAN
!
vlan 12
 description Ten1VLANNW1
!
```

```
Rack1-Device2# show running-config vlan
vlan 11
 description Tenant L2 Extended VLAN
!
vlan 12
 description Ten1VLANNW1
!
```

```
vlan 13
 router-interface Ve 13
 suppress-arp
 description Tenant L3 Extended VLAN
!
vlan 14
 router-interface Ve 14
 suppress-arp
 description Ten1VLANNW2
!
vlan 15
 description Tenant L3 Hand-off VLAN
!
vlan 16
 description Ten1VLANNW3
!

Rack1-Device1#
```

```
vlan 13
 router-interface Ve 13
 suppress-arp
 description Tenant L3 Extended VLAN
!
vlan 14
 router-interface Ve 14
 suppress-arp
 description Ten1VLANNW2
!
vlan 15
 description Tenant L3 Hand-off VLAN
!
vlan 16
 description Ten1VLANNW3
!
Rack1-Device2#
```

EPG: Update: anycast-ip-add or delete

The feature lets you add or delete anycast-ipv4 and anycast-ipv6 to or from an existing tenant network.

You can add or delete the anycast-ipv4 and anycast-ipv6 using the EPG update anycast-ip-add/delete operations on an L3 EPG.

You can provide ipv6 nd attributes (ipv6 nd mtu, ipv6 nd prefix, ipv6 nd m/o flags) along with anycast-ipv6 during anycast-ip-add operation.

You can provide only one anycast-ipv4 and one anycast-ipv6 per tenant network even though SLX supports multiple anycast-ipv4/ipv6 per VE.

Typical usage of the API from Openstack Integration is as follows:

1. EPG create with a ctag.
2. EPG update port-group-add with endpoints (po/phy).
3. EPG update vrf-add with anycast-ipv4.
4. EPG update anycast-ip-add with anycast-ipv6.

EPG Update: anycast-ip-add or delete

```
efa tenant epg update --name <epg-name> --tenant <tenant-name>
                --operation <anycast-ip-add | anycast-ip-delete>
                --anycast-ip <ctag:anycast-ipv4> --anycast-ipv6 <ctag:anycast-ipv6>
```

Example

```
efa tenant epg show --detail
================================================================================================
=====================================================================
Name          : ten1epg1
Tenant        : vlanTen1
Description   :
Type          : extension
Ports         : 10.20.246.15[0/1]
```

```
              : 10.20.246.16[0/1]
POs           :
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 11
              : vrf                 : ten1vrf1
              : l3-vni              : 8188
Network Property [Flags : * - Native Vlan]
+------+--------+--------------+--------------+---------+--------------------------------
+-------------------------+-------------+-----------------------+--------------------
+-------------+-------------+
| Ctag | L2-Vni | Anycast-IPv4 | Anycast-IPv6 | BD-name | Local IP (Device-IP->Local-IP)
|    Ctag-Description     | Mtu-IPv6-ND | ManagedConfig-IPv6-ND | OtherConfig-IPv6-ND |
Dev-state  |  App-state  |
+------+--------+--------------+--------------+---------+--------------------------------
+-------------------------+-------------+-----------------------+--------------------
+-------------+-------------+
| 11   | 11     | 10.0.11.1/24 |              |         |
| Tenant L3 Extended VLAN |              | False                 | False               |
provisioned | cfg-in-sync |
+------+--------+--------------+--------------+---------+--------------------------------
+-------------------------+-------------+-----------------------+--------------------
+-------------+-------------+
================================================================================================
=============================================================================

efa tenant epg update --name ten1epg1 --tenant vlanTen1 --operation anycast-ip-add --
anycast-ipv6 11:10::1/123

efa tenant epg show --detail
================================================================================================
=============================================================================
Name          : ten1epg1
Tenant        : vlanTen1
Description   :
Type          : extension
Ports         : 10.20.246.15[0/1]
              : 10.20.246.16[0/1]
POs           :
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 11
              : vrf                 : ten1vrf1
              : l3-vni              : 8188
Network Property [Flags : * - Native Vlan]
+------+--------+--------------+--------------+---------+--------------------------------
+-------------------------+-------------+-----------------------+--------------------
+-------------+-------------+
| Ctag | L2-Vni | Anycast-IPv4 | Anycast-IPv6 | BD-name | Local IP (Device-IP->Local-IP)
|    Ctag-Description     | Mtu-IPv6-ND | ManagedConfig-IPv6-ND | OtherConfig-IPv6-ND |
Dev-state  |  App-state  |
+------+--------+--------------+--------------+---------+--------------------------------
+-------------------------+-------------+-----------------------+--------------------
+-------------+-------------+
| 11   | 11     | 10.0.11.1/24 | 10::1/123    |         |
| Tenant L3 Extended VLAN |              | False                 | False               |
provisioned | cfg-in-sync |
+------+--------+--------------+------+---------+--------------------------------
+-------------------------+-------------+-----------------------+--------------------
+-------------+-------------+
================================================================================================
=============================================================================
```

Switch Config

```
Rack1-Device1# show running
-config interface Ve 11
interface Ve 11
 vrf forwarding ten1vrf1
 ip anycast-address 10.0.11.1/24
 ipv6 anycast-address 10::1/123
 no shutdown
!
Rack1-Device1#
```

```
Rack1-Device2# show running-config
interface Ve 11
interface Ve 11
 vrf forwarding ten1vrf1
 ip anycast-address 10.0.11.1/24
 ipv6 anycast-address 10::1/123
 no shutdown
!
Rack1-Device2#
```

EPG Update : anycast-ip-delete

```
efa tenant epg update --name ten1epg1 --tenant vlanTen1 --operation anycast-ip-delete --
anycast-ipv6 11:10::1/123

efa tenant epg show
================================================================================================
=====================================================================
Name          : ten1epg1
Tenant        : vlanTen1
Description   :
Type          : extension
Ports         : 10.20.246.15[0/1]
              : 10.20.246.16[0/1]
POs           :
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 11
              : vrf                 : ten1vrf1
              : l3-vni              : 8188
Network Property [Flags : * - Native Vlan]
+------+--------+--------------+--------------+---------+-------------------------------
+-------------+----------------------+--------------------+-------------+-------------+
| Ctag | L2-Vni | Anycast-IPv4 | Anycast-IPv6 | BD-name | Local IP (Device-IP->Local-IP)
| Mtu-IPv6-ND | ManagedConfig-IPv6-ND | OtherConfig-IPv6-ND |  Dev-state  |  App-state  |
+------+--------+--------------+--------------+---------+-------------------------------
+-------------+----------------------+--------------------+-------------+-------------+
| 11   | 11     | 10.0.11.1/24 |              |         |
|              | False                | False               | provisioned | cfg-in-sync |
+------+--------+--------------+--------------+---------+-------------------------------
+-------------+----------------------+--------------------+-------------+-------------+
For 'unstable' entities, run 'efa tenant po/vrf show' for details
================================================================================================
=====================================================================
```

Switch Config

```
Rack1-Device1# show running
-config interface Ve 11
interface Ve 11
 vrf forwarding ten1vrf1
 ip anycast-address 10.0.11.1/24
 no shutdown
!
Rack1-Device1#
```

```
Rack1-Device2# show running
-config interface Ve 11
interface Ve 11
 vrf forwarding ten1vrf1
 ip anycast-address 10.0.11.1/24
 no shutdown
!
Rack1-Device2#
```

EPG: Network Property: ipv6 nd

The feature lets you configure ipv6 nd attributes per tenant network (ctag). IPv6 ND attributes are MTU, M flag, O flag and Prefixes.

You can configure ipv6 nd attributes using EPG create or EPG update ctag-range-add, vrf-add, and anycast-ip-add operations.

EPG Create

```
efa tenant epg create --name <epg-name> --tenant <tenant-name>
               --ipv6-nd-mtu <ipv6-mtu>
               --ipv6-nd-managed-config <true | false>
               --ipv6-nd-other-config <true | false>
               --ipv6-nd-prefix <ctag:list-of-prefix>
               --ipv6-nd-prefix-valid-lifetime <ctag,prefix:validTime>
               --ipv6-nd-prefix-preferred-lifetime <ctag,prefix:preferredTime>
               --ipv6-nd-prefix-no-advertise <ctag,prefix:noadvertiseflag>
               --ipv6-nd-prefix-config-type <ctag,prefix:configType(no-autoconfig| no-
onlink | off-link)>
```

EPG Update

```
efa tenant epg update --name <epg-name> --tenant <tenant-name>
               --operation <ctag-range-add | vrf-add | anycast-ip-add> --ctag-range <ctag-
range> --vrf <vrf-name>
               --anycast-ip <ctag:anycast-ip> --anycast-ipv6 <ctag:anycast-ipv6>
               --ipv6-nd-mtu <ipv6-mtu>
               --ipv6-nd-managed-config <true | false>
               --ipv6-nd-other-config <true | false>
               --ipv6-nd-prefix <ctag:list-of-prefix>
               --ipv6-nd-prefix-valid-lifetime <ctag,prefix:validTime>
               --ipv6-nd-prefix-preferred-lifetime <ctag,prefix:preferredTime>
               --ipv6-nd-prefix-no-advertise <ctag,prefix:noadvertiseflag>
               --ipv6-nd-prefix-config-type <ctag,prefix:configType(no-autoconfig| no-
onlink | off-link)>
```

Example

```
efa tenant show
+----------+---------+------------+-------------+-------------+-----------+-----------
+----------------------+
|   Name   |  Type   | VLAN Range | L2VNI Range | L3VNI Range | VRF Count | Enable BD
|        Ports         |
+----------+---------+------------+-------------+-------------+-----------+-----------
+----------------------+
| vlanTen1 | private |   11-20    |             |             |    10     |   false   |
10.20.246.15[0/1-10] |
|          |         |            |             |             |           |           |
10.20.246.16[0/1-10] |
+----------+---------+------------+-------------+-------------+-----------+-----------
+----------------------+


efa tenant vrf show
+----------+----------+-------------+--------------------+-------------+----------
+-----------+-----------+--------------------+-------------+-------------+
|   Name   |  Tenant  | Routing Type | Centralized Routers | Redistribute | Max Path |
Local Asn | Enable GR |       State        |  Dev State  |  App State  |
+----------+----------+-------------+--------------------+-------------+----------
```

```
+-----------+-----------+--------------------+-------------+-------------+
| ten1vrf1 | vlanTen1 | distributed  |                     |  connected   |    8
|           |   false   | vrf-device-created | provisioned | cfg-in-sync |
+-----------+-----------+--------------------+-------------+-------------+----------
+-----------+-----------+--------------------+-------------+-------------+


efa tenant epg create --name ten1epg1 --tenant vlanTen1 --port
10.20.246.15[0/1],10.20.246.16[0/1] --switchport-mode trunk --ctag-range 11-13
           --anycast-ip 11:10.0.11.1/24 --anycast-ip 12:10.0.12.1/24 --anycast-ip
13:10.0.13.1/24
           --ipv6-nd-mtu 12:1600 --ipv6-nd-managed-config 12:true --ipv6-nd-other-config
12:true
           --ipv6-nd-prefix 12:1:5::/64 --ipv6-nd-prefix-valid-lifetime 12,1:5::/64:2000
--ipv6-nd-prefix-preferred-lifetime 12,1:5::/64:2000
           --ipv6-nd-prefix 12:1:6::/64 --ipv6-nd-prefix-valid-lifetime 12,1:6::/64:2001
--ipv6-nd-prefix-preferred-lifetime 12,1:6::/64:2001
           --vrf ten1vrf1

efa tenant epg show
+----------+----------+-----------+--------------------+----+------------+-------------
+------------+----------+-------+-------+
|   Name   |  Tenant  |   Type    |       Ports        | PO | SwitchPort | Native Vlan |
Ctag Range |   Vrf    | L3Vni | State |
|          |          |           |                    |    |    Mode    |   Tagging
|            |          |       |       |
+----------+----------+-----------+--------------------+----+------------+-------------
+------------+----------+-------+-------+
| ten1epg1 | vlanTen1 | extension | 10.20.246.15[0/1] |    |   trunk    |    false    |
11-13    | ten1vrf1 | 8192  |       |
|          |          |           | 10.20.246.16[0/1] |    |            |
|            |          |       |       |
+----------+----------+-----------+--------------------+----+------------+-------------
+------------+----------+-------+-------+


efa tenant epg show --detail

================================================================================================
=================================================================
Name          : ten1epg1
Tenant        : vlanTen1
Type          : extension
State         :
Description   :
Ports         : 10.20.246.15[0/1]
              : 10.20.246.16[0/1]
POs           :
Port Property : SwitchPort Mode     : trunk
              : Native Vlan Tagging : false
NW Policy     : Ctag Range          : 11-13
              : VRF                 : ten1vrf1
              : L3Vni               : 8192
+------+-------------------------+-------+---------+--------------+--------------
+----------------------+---------+----------------+--------------+-------------
+-------------+
| Ctag |          Ctag           | L2Vni | BD Name | Anycast IPv4 | Anycast IPv6 |
Local IP        | IPv6 ND |    IPv6 ND     |   IPv6 ND    |  Dev State  |  App State  |
|      |       Description       |       |         |              |              |
[Device-IP->Local-IP] |   Mtu   | Managed Config | Other Config |
|             |
+------+-------------------------+-------+---------+--------------+--------------
+----------------------+---------+----------------+--------------+-------------
+-------------+
```

```
|  11   | Tenant L3 Extended VLAN |  11   |         | 10.0.11.1/24 |
|                       |         |     false      |    false     | provisioned | cfg-in-
sync |
+------+-------------------------+-------+---------+--------------+--------------
+----------------------+---------+----------------+--------------+-------------
+-------------+
|  12   | Tenant L3 Extended VLAN |  12   |         | 10.0.12.1/24 |
|                       |  1600   |      true      |     true     | provisioned | cfg-in-
sync |
+------+-------------------------+-------+---------+--------------+--------------
+----------------------+---------+----------------+--------------+-------------
+-------------+
|  13   | Tenant L3 Extended VLAN |  13   |         | 10.0.13.1/24 |
|                       |         |     false      |    false     | provisioned | cfg-in-
sync |
+------+-------------------------+-------+---------+--------------+--------------
+----------------------+---------+----------------+--------------+-------------
+-------------+
Network Property [Flags : * - Native Vlan]
+------+----------------+-------------+----------------+--------------------
+-------------+
| Ctag | IPv6 ND Prefix | No Advertise | Valid Lifetime | Preferred Lifetime | Config
Type |
+------+----------------+-------------+----------------+--------------------
+-------------+
|  12   |    1:5::/64    |    false     |      2000      |        2000
|             |
+------+----------------+-------------+----------------+--------------------
+-------------+
|  12   |    1:6::/64    |    false     |      2001      |        2001
|             |
+------+----------------+-------------+----------------+--------------------
+-------------+
IPv6 ND Prefix Flags
For 'unstable' entities, run 'efa tenant po/vrf show' for details
================================================================================================
=================================================================
```

Switch Config

```
Rack1-Device1# show running
-config interface Ve 12
interface Ve 12
 vrf forwarding ten1vrf1
 ip anycast-address 10.0.12.1/24
 ipv6 nd managed-config-flag
 ipv6 nd other-config-flag
 ipv6 nd mtu 1600
 ipv6 nd prefix 1:5::/64 2000 2000
 ipv6 nd prefix 1:6::/64 2001 2001
 no shutdown
!
Rack1-Device1#
```

```
Rack1-Device2# show running
-config interface Ve 12
interface Ve 12
 vrf forwarding ten1vrf1
 ip anycast-address 10.0.12.1/24
 ipv6 nd managed-config-flag
 ipv6 nd other-config-flag
 ipv6 nd mtu 1600
 ipv6 nd prefix 1:5::/64 2000 2000
 ipv6 nd prefix 1:6::/64 2001 2001
 no shutdown
!
Rack1-Device2#
```

Configure the BFD Session Type for an Endpoint Group

You can determine the session type for a BFD session formed over a CEP port.

You can assign a session type as you create an endpoint group. You can also assign a session type to an existing endpoint group.

The default is auto, which means that the BFD session type is automatically determined based on whether the service type is set to extension (software) or Layer 3 hand-off (hardware).

The value of `--single-homed-bfd-session-type` is configured for one endpoint group and then propagated to all Ethernet and single-homed port channel interfaces defined for that endpoint group.

EFA does not distinguish between SRIOV (single-root input/output virtualization) and non-SRIOV connections. Therefore, it treats both connections the same way. If you want to use hardware-based BFD sessions for CEP non-SRIOV connections, then create an endpoint group that contains all the CEP non-SRIOV connections and set the `--single-homed-bfd-session-type` to `hardware`.

1. To assign a BFD session type as you create an endpoint group, run the following command.

```
$ efa tenant epg create --name epg5 --tenant tenant11 --port 10.20.216.15[0/11]
,10.20.216.16[0/11] --po po1 --switchport-mode trunk --single-homed-bfd-session-type
auto
```

   In this example, the session type is set to 'auto'.

2. To assign a BFD session type to an existing endpoint group, run the following command.

```
$ efa tenant epg update --name epg5 --tenant tenant11 --operation port-group-add
--port 10.20.216.15[0/11],10.20.216.16[0/11] --po po1 --switchport-mode trunk
--single-homed-bfd-session-type hardware
```

   In this example, the session type is set to 'hardware'.

EPG: CEP Cluster Tracking

EFA does not provision `reload-delay 90` configuration on Cluster Edge Port (CEP) interfaces. EFA instead provisions `cluster-track` configuration on CEP interfaces when the CEP is configured as a member of an EPG (endpoint group) during the creation or updating of endpoint groups.

During upgrade from EFA 2.2.0 to 2.3.0, EFA marks all the CEPs with the intended `cluster-track` configuration and shows it as configuration drift. On reconciliation of the drift, EFA pushes the `cluster-track` configuration to the CEP ports. Before the `cluster-track` configuration push, EFA automatically removes the `reload-delay` configuration from the CEP ports.

EFA Provisioning

```
# efa tenant po create --name po11 --tenant tenant1 -speed 10Gbps --negotiation active
--port 10.24.80.134[0/15] => CEP # efa tenant po create --name po12 --tenant tenant1
--speed 10Gbps --negotiation active --port 10.24.80.134[0/25],10.24.80.135[0/25] => CCEP

# efa tenant epg create --name ten1epg1 --tenant tenant1 --switchport-mode trunk --ctag-
range 1001
--port 10.24.80.134[0/35] --po po11,po12
```

Enable Cluster Tracking on CEP Interfaces

By default, EFA enables reload-delay configuration on all Cluster Edge Port (CEP) Interfaces. Reload-delay and cluster-tracking configurations are mutually exclusive.

When cluster tracking is enabled, an interface can track the state of an MCT (multi-chassis tunnel) cluster and divert traffic to alternative paths when a cluster is down for reasons such as maintenance mode.

To enable cluster tracking on CEP interfaces, you must remove the reload-delay configuration.

1. Remove the reload-delay configuration on CEP interface.

```
# efa inventory device execute-cli --ip 10.18.120.187 --command "Interface ethernet
0/1, no reload-delay enable" --config
```

2. Configure cluster tracking.

```
# efa inventory device execute-cli --ip 10.18.120.187 --command "Interface ethernet
0/1, cluster-track" --config
```

EPG: Local IP

You can add and delete local IP address configurations.

You can add and delete local IP address configurations during the following operations:

- Creating EPGs
- Adding or deleting CTAG ranges
- Adding or deleting VRFs

The Local IP address is configured on the VE interface assigned to a particular tenant network. You can select different local IP addresses for each device in a tenant network.

EPG create with local IP configuration

```
efa tenant epg create --name ten1epg1 --tenant tenant1 --vrf red
--switchport-mode trunk --ctag-range 11 --anycast-ip 11:10.10.11.1/24
--port 10.24.80.150[0/1],10.24.80.151[0/1]
--local-ip 11,10.24.80.150:11.22.33.41/24 --local-ip 11,10.24.80.151:11.22.34.41/24

efa tenant epg show
Name: ten1epg1
Tenant: tenant1
Description:
Type: extension
Ports : 10.24.80.151[0/1]
      : 10.24.80.150[0/1]
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy: ctag-range       :11
              : vrf                 : red
              : vrf-State           : vrf-device-created
              : vrf-Device-State    : provisioned
              : vrf-App-State       : cfg-refreshed
              : l3-vni              : 8190
Network Property [Flags : * - Native Vlan]

| Ctag | L2-Vni | Anycast-IPv4 | Anycast-IPv6 | BD-name |                Local IP (Device-
IP->Local-IP)                |  Dev-state  |    App-state    |
+------+--------+--------------+--------------+---------
+-------------------------------------------------------------+-------------
+-----------------+
| 11   | 11     |10.10.11.1/24 |              |         |    10.24.80.151-
>11.22.34.41/24                             | provisioned | cfg-refreshed   |
|      |        |              |              |
|                                                              |
|                  |
|      |        |              |              |         |    10.24.80.150-
>11.22.33.41/24                             |             |                 |
```

EPG update local-ip-delete operation

```
efa tenant epg update --name epgv20 --tenant tenant1 --operation
local-ip-delete --local-ip 11,10.24.80.150:11.22.33.41/24

efa tenant epg show
Name          : epgv20
Tenant        : t3
Description   :
Type          : l3-hand-off
Ports         : 10.20.50.209[0/27]
POs           : posv9
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 201-202
              : vrf                 : vrfv20
              : l3-vni              : 5110

Network Property [Flags : * - Native Vlan]
+------+--------+--------------+--------------+---------
+-------------------------------------------------------------------+-------------
+-----------------+
| Ctag | L2-Vni | Anycast-IPv4 | Anycast-IPv6 | BD-name |               Local IP (Device-
IP->Local-IP)                |  Dev-state  |    App-state    |
+------+--------+--------------+--------------+---------
+-------------------------------------------------------------------+-------------
+-----------------+
| 201  | 201    |              |              |         |    10.20.50.209-
>44.4.4.5/24                                | provisioned | cfg-in-sync     |
|      |        |              |              |         |
4444:44::5/120                              |             |                 |
|      |        |              |              |         |    10.20.50.208-
>44.4.4.4/24                                |             |                 |
|      |        |              |              |         |
4444:44::4/120                              |             |                 |
+------+--------+--------------+--------------+---------
+-------------------------------------------------------------------+-------------
+-----------------+
| 202  | 202    |              |              |         |    10.20.50.209-
>44.4.5.5/24                                | provisioned | cg-in-sync      |
|      |        |              |              |         |
4444:45::5/120                              |             |                 |
+------+--------+--------------+--------------+---------
+-------------------------------------------------------------------+-------------
+-----------------+

For 'unstable' entities, run 'efa tenant po/vrf show' for details
================================================================================================
=================================================================
```

EPG update local-ip-add operation

```
efa tenant epg update --name ten1epg1 --tenant tenant1
--operation local-ip-add --local-ip 11,10.24.80.150:11.22.33.41/24

efa tenant epg show
Name: ten1epg1
Tenant: tenant1
Description:
Type: extension
Ports : 10.24.80.151[0/1]
      : 10.24.80.150[0/1]
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
```

```
NW Policy: ctag-range       :11
              : vrf                 : red
              : vrf-State           : vrf-device-created
              : vrf-Device-State    : provisioned
              : vrf-App-State       : cfg-refreshed
              : l3-vni              : 8190

Network Property [Flags : * - Native Vlan]
+------+--------+--------------+--------------+---------
+-----------------------------------------------------------------+-------------
+-----------------+
| Ctag | L2-Vni | Anycast-IPv4 | Anycast-IPv6 | BD-name |                Local IP (Device-
IP->Local-IP)                |  Dev-state  |    App-state    |
+------+--------+--------------+--------------+---------
+-----------------------------------------------------------------+-------------
+-----------------+
| 11   | 11     |              |              |         |  10.24.80.151-
>11.22.34.41/24                                | provisioned | cfg-in-sync     |
|      |        |              |              |
|                                                                  |
|                  |
|      |        |              |              |         |    10.24.80.150-
>11.22.33.41/24                               |             |
```

BGP as a Service

BGP as a service allows creation and deletion of BGP peer and peer-group neighbors on a given fabric device. The VRF must be created on the fabric device using EPG create/update prior to the BGP neighbors.

BGP Peer Group

BGP Peer Group Instance Create

```
# efa tenant service bgp peer-group create --name <peer-group-name> --tenant <tenant-
name> --description <description> --pg-name <switch-ip:pg-name> --pg-asn <switch-ip:pg-
name,remote-asn> --pg-bfd <switch-ip:pg-name,bfd-enable(true/false),interval,min-
rx,multiplier> --pg-next-hop-self <switch-ip:pg-name,next-hop-self(true/false)> --pg-
update-source-ip <switch-ip:pg-name,update-source-ip>
```

```
# efa tenant service bgp peer-group create -name ten1BgpPG1 --tenant tenant1 --pg-name
10.24.80.134:pg1 --pg-asn 10.24.80.134:pg1,6000 --pg-bfd 10.24.80.134:pg1,true,100,200,5
--pg-next-hop-self 10.24.80.134:pg1,true --pg-update-source-ip
10.24.80.134:pg1,10.20.30.40
```

BGP Peer Group Instance Update - Peer Group Add

```
# efa tenant service bgp peer-group update --name <peer-group-name> --tenant <tenant-
name> --operation <peer-group-add|peer-group-delete|peer-group-desc-update> --description
<description> --pg-name <switch-ip:pg-name> --pg-asn <switch-ip:pg-name,remote-asn> --pg-
bfd <switch-ip:pg-name,bfd-enable(true/false),interval,min-rx,multiplier> --pg-next-hop-
self <switch-ip:pg-name,next-hop-self(true/false)> --pg-update-source-ip <switch-ip:pg-
name,update-source-ip>
```

```
efa tenant service bgp peer-group update --name ten1BgpPG1 --tenant tenant1 --operation
peer-group-add --pg-name 10.24.80.134:pg2 -pg-asn 10.24.80.134:pg2,7000 --pg-bfd
10.24.80.134:pg2,true,200,300,6 --pg-next-hop-self 10.24.80.134:pg2,true --pg-update-
source-ip 10.24.80.134:pg2,10.20.30.41
```

BGP Peer Group Instance Show

```
efa tenant service bgp peer-group show
================================================================================================
=======================================
Name : ten1BgpPG1
Tenant : tenant1
State : bs-state-created
+--------------+-------------+------------+-------------+--------------+--------
+----------------+-------------+-------------+
| Device IP | PeerGroup | REMOTE ASN | BFD Enabled | BFD Interval | BFD Rx | BFD
Multiplier | Dev-state | App-state |
+--------------+-------------+------------+-------------+--------------+--------
+----------------+-------------+-------------+
| 10.24.80.134 | pg1 | 6000 | true | 100 | 200 | 5 | provisioned | cfg-in-sync |
+--------------+-------------+------------+-------------+--------------+--------
+----------------+-------------+-------------+
| 10.24.80.134 | pg2 | 7000 | true | 200 | 300 | 6 | provisioned | cfg-in-sync |
+--------------+-------------+------------+-------------+--------------
+-------------------------+-------------+-------------+
================================================================================================
=======================================
```

Switch Config

```
Rack1-Device1# show running-config router bgp
router bgp
local-as 100
neighbor pg1 peer-group
neighbor pg1 remote-as 6000
neighbor pg1 update-source 10.20.30.40
neighbor pg1 next-hop-self neighbor pg1 bfd
neighbor pg1 bfd interval 100 min-rx 200 multiplier 5
neighbor pg2 peer-group
neighbor pg2 remote-as 7000
neighbor pg2 update-source 10.20.30.41
neighbor pg2 next-hop-self neighbor pg2 bfd
neighbor pg2 bfd interval 200 min-rx 300 multiplier 6
address-family ipv4 unicast
!
address-family ipv6 unicast
!
address-family l2vpn evpn
!
```

BGP Peer Group Instance Update - Peer Group Delete

```
# efa tenant service bgp peer-group update --name ten1BgpPG1 --tenant tenant1 --operation
peer-group-delete --pg-name 10.24.80.134:pg2
```

```
# efa tenant service bgp peer-group show
================================================================================================
=======================================
Name : ten1BgpPG1
Tenant : tenant1
State : bs-state-created
+--------------+-------------+------------+-------------+--------------+--------
+----------------+-------------+-------------+
| Device IP | PeerGroup | REMOTE ASN | BFD Enabled | BFD Interval | BFD Rx | BFD
Multiplier | Dev-state | App-state |
+--------------+-------------+------------+-------------+--------------+--------
+----------------+-------------+-------------+
| 10.24.80.134 | pg1 | 6000 | true | 100 | 200 | 5 | provisioned | cfg-in-sync |
+--------------+-------------+------------+-------------+--------------
```

```
+------------------------+-------------+-------------+
================================================================================================
=========================================
```

BGP Peer Group Instance Delete

```
# efa tenant service bgp peer-group delete --name ten1BgpPG1 --tenant tenant1
```

BGP Peer

BGP Peer Instance Create - Static Peer

```
# efa tenant service bgp peer create --name <peer-name> --tenant <tenant-name> --ipv4-uc-
nbr <switch-ip,vrf-name:ipv4-neighbor,remote-as> --ipv4-uc-nbr-bfd <switch-ip,vrf-
name:ipv4-neighbor,bfd-enable(true/false),bfd-interval,bfd-rx,bfd-mult> --ipv4-uc-nbr-
update-source-ip <switch-ip,vrf-name:ipv4-neighbor,update-source-ip> --ipv4-uc-nbr-next-
hop-self <switch-ip,vrf-name:ipv4-neighbor,next-hop-self(true/false)> --ipv6-uc-nbr
<switch-ip,vrf-name:ipv6-neighbor,remote-as> --ipv6-uc-nbr-bfd <switch-ip,vrf-name:ipv6-
neighbor,bfd-enable(t/f),bfd-interval,bfd-rx,bfd-mult> --ipv6-uc-nbr-update-source-ip
<switch-ip,vrf-name:ipv6-neighbor,update-source-ip> --ipv6-uc-nbr-next-hop-self <switch-
ip,vrf-name:ipv6-neighbor,next-hop-self(true/false)>
```

```
# efa tenant service bgp peer create --name bgpservice1 --tenant tenant1 --ipv4-uc-nbr
10.24.80.134,red:10.20.30.40,5000 --ipv4-uc-nbr-bfd
10.24.80.134,red:10.20.30.40,true,100,200,5 --ipv4-uc-nbr-update-source-ip
10.24.80.134,red:10.20.30.40,11.22.20.33 --ipv4-uc-nbr-next-hop-self
10.24.80.134,red:10.20.30.40,true
```

BGP Peer Instance Show - Static Peer

```
# efa tenant service bgp show
```

```
# efa tenant service bgp peer show
================================================================================================
================================================================
Name : bgpservice1
Tenant : tenant1
State : bs-state-created
+--------------+-----+------+---------+-------------+------------+-------------
+--------------+--------+---------------+-------------+-------------+
| Device IP | VRF | AFI | SAFI | REMOTE IP | REMOTE ASN | BFD Enabled | BFD Interval |
BFD Rx | BFD Multiplier | Dev-state | App-state |
+--------------+-----+------+---------+-------------+------------+-------------
+--------------+--------+---------------+-------------+-------------+
| 10.24.80.134 | red | ipv4 | unicast | 10.20.30.40 | 5000 | true | 100 | 200 | 5 |
provisioned | cfg-in-sync |
+--------------+-----+------+---------+-------------+------------+-------------
+--------------+--------+---------------+-------------+-------------+
================================================================================================
================================================================
```

BGP Peer Instance Update - Static Peer Add

```
# efa tenant service bgp peer update --name <peer-name> --tenant <tenant-name> --
operation peer-add --ipv4-uc-nbr <switch-ip,vrf-name:ipv4-neighbor,remote-as> --ipv4-uc-
nbr-bfd <switch-ip,vrf-name:ipv4-neighbor,bfd-enable(t/f),bfd-interval,bfd-rx,bfd-mult> --
ipv4-uc-nbr-update-source-ip <switch-ip,vrf-name:ipv4-neighbor,update-source-ip> --ipv4-
uc-nbr-next-hop-self <switch-ip,vrf-name:ipv4-neighbor,next-hop-self(true/false)> --ipv6-
uc-nbr <switch-ip,vrf-name:ipv6-neighbor,remote-as> --ipv6-uc-nbr-bfd <switch-ip,vrf-
name:ipv6-neighbor,bfd-enable(t/f),bfd-interval,bfd-rx,bfd-mult> --ipv6-uc-nbr-update-
```

```
source-ip <switch-ip,vrf-name:ipv6-neighbor,update-source-ip> --ipv6-uc-nbr-next-hop-self
<switch-ip,vrf-name:ipv6-neighbor,next-hop-self(true/false)>
```

```
# efa tenant service bgp peer update --name bgpservice1 --tenant tenant1 --operation peer-
add --ipv6-uc-nbr 10.24.80.134,red:10::40,5000 --ipv6-uc-nbr-bfd
10.24.80.134,red:10::40,true,100,200,5 --ipv6-uc-nbr-update-source-ip
10.24.80.134,red:10::40,11::22 --ipv6-uc-nbr-next-hop-self 10.24.80.134,red:10::40,true
efa tenant service bgp peer show
================================================================================================
================================================================
Name : bgpservice1
Tenant : tenant1
State : bs-state-created
+--------------+-----+------+---------+-------------+------------+-------------
+--------------+--------+----------------+-------------+-------------+
| Device IP | VRF | AFI | SAFI | REMOTE IP | REMOTE ASN | BFD Enabled | BFD Interval |
BFD Rx | BFD Multiplier | Dev-state | App-state |
+--------------+-----+------+---------+-------------+------------+-------------
+--------------+--------+----------------+-------------+-------------+
| 10.24.80.134 | red | ipv6 | unicast | 10::40 | 5000 | false | 100 | 200 | 5 |
provisioned | cfg-in-sync |
+--------------+-----+------+---------+-------------+------------+-------------
+--------------+--------+----------------+-------------+-------------+
| 10.24.80.134 | red | ipv4 | unicast | 10.20.30.40 | 5000 | false | 100 | 200 | 5 |
provisioned | cfg-in-sync |
+--------------+-----+------+---------+-------------+------------+-------------
+--------------+--------+----------------+-------------+-------------+
================================================================================================
================================================================
```

Switch Config

```
Rack1-Device1# sh run router bgp
router bgp
local-as 100
address-family ipv4 unicast
!
address-family ipv4 unicast vrf red
neighbor 10.20.30.40 remote-as 5000
neighbor 10.20.30.40 next-hop-self
neighbor 10.20.30.40 bfd
neighbor 10.20.30.40 bfd interval 100 min-rx 200 multiplier 5
neighbor 10.20.30.40 update-source 11.22.20.33
!
address-family ipv6 unicast
!
address-family ipv6 unicast vrf red
neighbor 10::40 remote-as 5000
neighbor 10::40 next-hop-self
neighbor 10::40 bfd
neighbor 10::40 bfd interval 100 min-rx 200 multiplier 5
neighbor 10::40 update-source 11::22
!
address-family l2vpn evpn
!
```

BGP Peer Instance Update - Static Peer Delete

```
# efa tenant service bgp peer update --name <peer-name> --tenant <tenant-name> --
operation peer-delete --ipv4-unicast-neighbor <switch-ip,vrf-name:ipv4-neighbor> --ipv6-
unicast-neighbor <switch-ip,vrf-name:ipv4-neighbor >
```

```
# efa tenant service bgp peer update --name bgpservice1 --tenant tenant1 --operation peer-
delete --ipv4-unicast-neighbor 10.24.80.134,red:10.20.30.40
# efa tenant service bgp show
================================================================================================
================================================================
Name : bgpservice1
Tenant : tenant1
State : bs-state-created
+--------------+-----+------+---------+-------------+------------+-------------
+--------------+--------+----------------+-------------+-------------+
| Device IP | VRF | AFI | SAFI | REMOTE IP | REMOTE ASN | BFD Enabled | BFD Interval |
BFD Rx | BFD Multiplier | Dev-state | App-state |
+--------------+-----+------+---------+-------------+------------+-------------
+--------------+--------+----------------+-------------+-------------+
| 10.24.80.134 | red | ipv6 | unicast | 10::40 | 5000 | false | 0 | 0 | 0 | provisioned |
cfg-in-sync |
+--------------+-----+------+---------+-------------+------------+-------------
+--------------+--------+----------------+-------------+-------------+
================================================================================================
================================================================
```

BGP Peer Instance Create - Dynamic Peer

```
# efa tenant service bgp peer create --name <peer-name> --tenant <tenant-name> --ipv4-uc-
dyn-nbr <switch-ip,vrf-name:listen-range,peer-group-name,listen-limit> --ipv6-uc-dyn-nbr
<switch-ip,vrf-name:listen-range,peer-group-name,listen-limit>
```

```
# efa tenant service bgp peer create --name bgpservice1 --tenant tenant1 --operation peer-
add –ipv4-uc-dyn-nbr 10.24.80.134,red:11::22/127,pg1,20
```

BGP Peer Instance Show - Dynamic Peer

```
# efa tenant service bgp peer show
================================================================================================
================================================================
Name : bgpservice1
Tenant : tenant1
State : bs-state-created
+--------------+-----+------+---------+----------------+------------+-------------
+-------------+-------------+
| Device IP | VRF | AFI | SAFI | LISTEN RANGE | Peer Group | LISTEN LIMIT | Dev-state |
App-state |
+--------------+-----+------+---------+----------------+------------+-------------
+-------------+-------------+
| 10.24.80.134 | red | ipv4 | unicast | 11.22.33.44/30 | pg1 | 10 | provisioned | cfg-in-
sync |
+--------------+-----+------+---------+----------------+------------+-------------
+-------------+-------------+ | 10.24.80.134 | red | ipv6 | unicast | 11::22/127 | pg1 |
20 | provisioned | cfg-in-sync |
+--------------+-----+------+---------+----------------+------------+-------------
+-------------+-------------+
================================================================================================
================================================================
```

Switch Config

```
Rack1-Device1# sh run router bgp
router bgp
local-as 100
neighbor pg1 peer-group
neighbor pg1 remote-as 6000
address-family ipv4 unicast
!
address-family ipv4 unicast vrf red
listen-range 11.22.33.44/30 peer-group pg1 limit 10
!
address-family ipv6 unicast
!
address-family ipv6 unicast vrf red
listen-range 11::22/127 peer-group pg1 limit 20
!
address-family l2vpn evpn
!
!
```

BGP Peer Instance Update - Dynamic Peer Delete

```
# efa tenant service bgp peer update --name <peer-name> --tenant <tenant-name> --
operation peer-delete --ipv4-uc-dyn-nbr <switch-ip,vrf-name:listen-range,peer-group-name>
--ipv6-uc-dyn-nbr <switch-ip,vrf-name:listen-range,peer-group-name>
```

```
# efa tenant service bgp peer create --name bgpservice1 --tenant tenant1 --operation peer-
delete –ipv4-uc-dyn-nbr 10.24.80.134,red:11::22/127,pg1

# efa tenant service bgp peer show
================================================================================================
================================================================
Name : bgpservice1
Tenant : tenant1
State : bs-state-created
+--------------+-----+------+---------+----------------+------------+--------------
+-------------+-------------+
| Device IP | VRF | AFI | SAFI | LISTEN RANGE | Peer Group | LISTEN LIMIT | Dev-state |
App-state |
+--------------+-----+------+---------+----------------+------------+--------------
+-------------+-------------+
| 10.24.80.134 | red | ipv4 | unicast | 11.22.33.44/30 | pg1 | 10 | provisioned | cfg-in-
sync |
+--------------+-----+------+---------+----------------+------------+--------------
+-------------+-------------+
================================================================================================
================================================================
```

BGP Peer Instance Delete

```
efa tenant service bgp peer delete --name <peer-name> --tenant <tenant-name>
```

```
# efa tenant service bgp peer delete –name bgpservice1 –tenant tenant1
```

IPv6 Support

EFA supports the provisioning of IPv6 anycast gateways.

EFA provisioning

The following commands support anycast gateways.

```
efa tenant vrf create --name vrf1 --tenant tenant1 --rt-type both --rt 1:1

efa tenant epg create --name ten1epg1 --tenant tenant1 --port 10.24.80.134[0/15]
--switchport-mode trunk --ctag-range 1001 --anycast-ip 1001:10.0.1.1/24
--anycast-ipv6 1001:2001:10:0:1::1/64 --vrf vrf1
```

The following examples show the configuration on the related SLX devices after configuration in EFA.

Table 13: Related SLX device configuration

vlan 1001 router-interface Ve 1001 **suppress-nd** suppress-arp ! leaf-9250-173# sh run in ve 1001 interface Ve 1001 vrf forwarding vrf1 ip anycast-address 10.0.1.1/24 **ipv6 anycast-address** **2001:10:0:1::1/64** no shutdown !	leaf-9250-173# sh run vrf vrf1 vrf vrf1 rd 172.31.254.13:1 evpn irb ve 8192 address-family ipv4 unicast route-target export 1:1 evpn route-target import 1:1 evpn ! **address-family ipv6 unicast** **route-target export 1:1 evpn** **route-target import 1:1 evpn** **!** !
leaf-9250-173# show running-config router bgp address-family ipv4 unicast vrf vrf1 router bgp address-family ipv4 unicast vrf vrf1 local-as 4210000001 maximum-paths 2 !	leaf-9250-173# show running-config router bgp address-family ipv6 unicast vrf vrf1 router bgp **address-family ipv6 unicast vrf vrf1** **redistribute connected** **maximum-paths 2** !

Exclusion of VLANs and Bridge from Cluster Instance

EFA excludes the VLANs and bridge domains used in the Layer 3 hand-off (toward the external gateway) endpoint group from the cluster instance by configuring `member vlan remove <vlan-range>` and `member bridge-domain remove <bd-range>` under the cluster instance.

During EFA upgrade, EFA marks all the VLANs/BDs used in `l3-hand-off` EPGs with the intended `member vlan remove <vlan-range>` and `member bridge-domain remove <bd-range>` configuration and shows as configuration drift. On reconciliation of the drift, EFA pushes `member`

`vlan remove <vlan-range>` and `member bridge-domain remove <bd-range>` configuration under the cluster.

EFA Provisioning

```
# efa tenant create --name tenant1 --port 10.24.80.134[0/1-10],10.24.80.135[0/1-10]
--vlan-range 2001-2010

# efa tenant po create --name po1 --tenant tenant1 --port
10.24.80.134[0/1],10.24.80.135[0/1]
--speed 10Gbps --negotiation active

# efa tenant epg create --name L3HandoffEPG1Ten1 --tenant tenant1 --ctag-range 2001-2003
--switchport-mode trunk --po po1 --type l3-hand-off
```

```
Device1 # show run interface Port-channel 1

interface Port-channel 1
cluster-client auto
switchport
switchport mode trunk
switchport trunk allowed vlan add 2001-2003
no switchport trunk tag native-vlan
no shutdown
!
Device1# show running config-evpn
evpn-fabric1
route-target both auto ignore-as
rd auto
duplicate-mac-timer 5 max-count 3
!
Device1# show running-config cluster
cluster fabric1-cluster-1
peer 10.20.20.5
peer-interface Port-channel 64
peer-keepalive
auto
!
member vlan-all
member vlan remove 2001-2003
member bridge-domain all
!
```

```
Device2 # show run interface Port-channel 1

interface Port-channel 1
cluster-client auto
switchport
switchport mode trunk
switchport trunk allowed vlan add 2001-2003
no switchport trunk tag native-vlan
no shutdown
!
Device2# show running config-evpn
evpn-fabric1
route-target both auto ignore-as
rd auto
duplicate-mac-timer 5 max-count 3
!
Device2# show running-config cluster
cluster fabric1-cluster-1
peer 10.20.20.5
peer-interface Port-channel 64
peer-keepalive
auto
```

```
!
member vlan-all
member vlan remove 2001-2003
member bridge-domain all
!
```

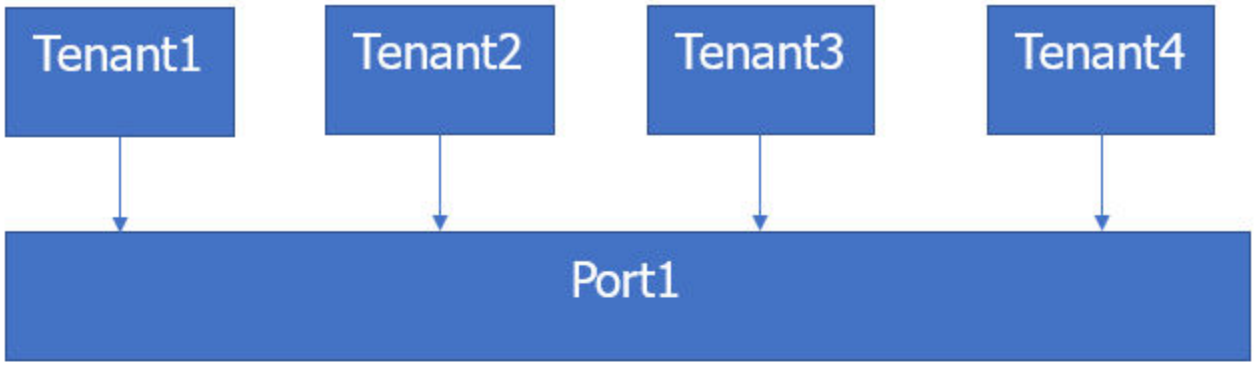
Sharing Resources Across Tenants using Shared Tenant

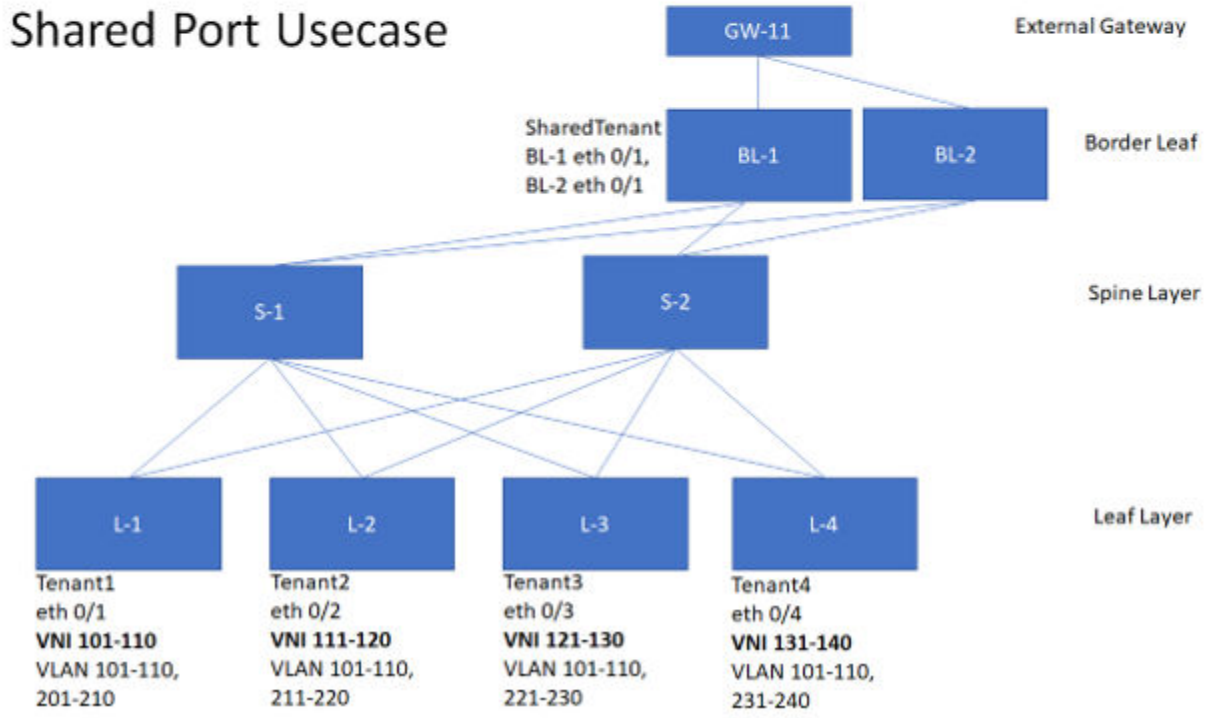
This topic offers examples of configuring shared tenant resources.

One tenant, with the `role=shared` attribute, owns the resources and entities that can be shared across all the other tenants, called non-shared tenant. The tenant service can have one shared tenant that services all the shared resources. The shared tenant owns the physical ports, Layer 2 and Layer 3 VNI number ranges, VLAN number ranges, and the VRF numbers. The shared tenant can create the endpoints and the VRFs, but not the endpoint groups.

A non-shared tenant cannot use the ports that the shared tenant owns if the ports are already part of an endpoint. A non-shared tenant cannot create an endpoint using the ports that the shared tenant owns.

Example: Shared port use case (Layer 2 hand-off)





The following examples show the commands and syntax used to configure the shared port.

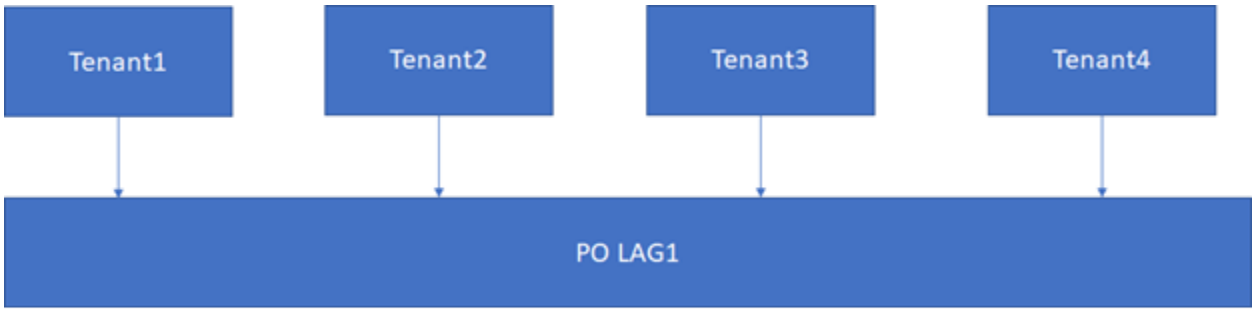
```
efa tenant create --name tenant1 --l2-vni-range 101-110 --vlan-range 101-110,201-210
--port L-1[0/1]
efa tenant create --name tenant2 --l2-vni-range 111-120 --vlan-range 101-110,211-220
--port L-2[0/2]
efa tenant create --name tenant3 --l2-vni-range 121-130 --vlan-range 101-110,221-230
--port L-3[0/3]
efa tenant create --name tenant4 --l2-vni-range 131-140 --vlan-range 101-110,231-240
--port L-4[0/4]
```

```
efa tenant create --name SharedTenant --port BL-1[0/1],BL-2[0/1] --type shared
```

```
efa tenant epg create --name ten1epg1 --tenant tenant1 --port L-1[0/1]
--switchport-mode trunk --ctag-range 101-110 --l2-vni 101:101 --l2-vni 102:102 …..
--l2-vni 110:110

efa tenant epg create --name ten2epg1 --tenant tenant2 --port L-2[0/2]
--switchport-mode trunk --ctag-range 101-110 --l2-vni 101:111 --l2-vni 102:112 …..
--l2-vni 110:120

efa tenant epg create --name ten3epg1 --tenant tenant3 --port L-3[0/3]
--switchport-mode trunk --ctag-range 101-110 --l2-vni 101:121 --l2-vni 102:122 …..
--l2-vni 110:130

efa tenant epg create --name ten4epg1 --tenant tenant4 --port L-4[0/4]
--switchport-mode trunk --ctag-range 101-110 --l2-vni 101:131 --l2-vni 102:132 …..
--l2-vni 110:140
```

```
efa tenant epg create --name ten1epg2 --tenant tenant1 --port BL-1[0/1],BL-2[0/1]
--switchport-mode trunk --ctag-range 201-210 --l2-vni 201:101 --l2-vni 202:102 …..
--l2-vni 210:110

efa tenant epg create --name ten2epg2 --tenant tenant2 --port BL-1[0/1],BL-2[0/1]
--switchport-mode trunk --ctag-range 211-220 --l2-vni 211:111 --l2-vni 212:112 …..
--l2-vni 220:120

efa tenant epg create --name ten3epg2 --tenant tenant3 --port BL-1[0/1],BL-2[0/1]
--switchport-mode trunk --ctag-range 221-230 --l2-vni 221:121 --l2-vni 212:122 …..
--l2-vni 230:130

efa tenant epg create --name ten4epg2 --tenant tenant4 --port BL-1[0/1],BL-2[0/1]
--switchport-mode trunk --ctag-range 231-240 --l2-vni 231:131 --l2-vni 212:132 …..
--l2-vni 240:140
```

Example: Shared endpoint use case (Layer 2 hand-off)



The following examples show the commands and syntax used to configure the shared endpoint.

```
efa tenant create --name SharedTenant --port BL-1[0/1],BL-2[0/1]
--l3-vni-range 1001-1010 --vrf-count 10 --type shared efa tenant vrf create --name red
--tenant SharedTenant

efa tenant epg create --name ten1epg1 --tenant tenant1 --port L-1[0/1]
--switchport-mode trunk --ctag-range 101-102 --l2-vni 101:101 --l2-vni 102:102
--anycast-ip 101:10.10.10.1/24 --vrf red --l3-vni 1001

efa tenant epg create --name ten2epg1 --tenant tenant2 --port L-2[0/2]
--switchport-mode trunk --ctag-range 101-102 --l2-vni 101:111 --l2-vni 102:112
--anycast-ip 101:10.10.11.1/24 --vrf red --l3-vni 1001

efa tenant epg create --name ten3epg1 --tenant tenant3 --port L-3[0/3]
--switchport-mode trunk --ctag-range 101-102 --l2-vni 101:121 --l2-vni 102:122
--anycast-ip 101:10.10.12.1/24 --vrf red --l3-vni 1001

efa tenant epg create --name ten1epg1 --tenant tenant4 --port L-4[0/4]
--switchport-mode trunk --ctag-range 101-102 --l2-vni 101:131 --l2-vni 102:132
--anycast-ip 101:10.10.13.1/24 --vrf red --l3-vni 1001
```

Example: Shared endpoint use case (Layer 3 hand-off)

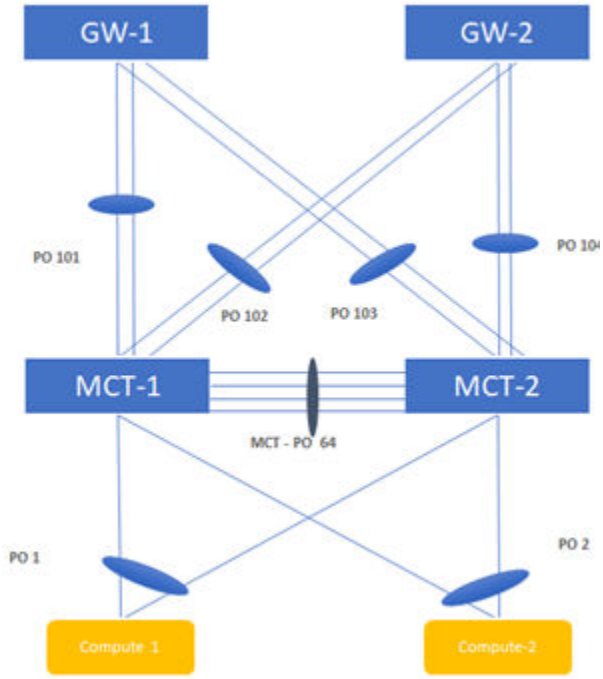


Figure 19: Topology

The following examples show the commands and syntax used to configure the shared endpoint.

```
efa tenant create --name tenant1 --l2-vni-range 1001-1010 --vlan-range 1001-1010
--port BL-1[0/11],BL-2[0/11] --l3-vni-range 10001-10010 --vrf-count 10

efa tenant create --name tenant2 --l2-vni-range 1101-1110 --vlan-range 1101-1110
--port BL-1[0/21],BL-2[0/21] --l3-vni-range 20001-20010 --vrf-count 10
```

```
efa tenant vrf create --name vrf1 --tenant Tenant1

efa tenant vrf create --name vrf2 --tenant Tenant2
```

```
efa tenant epg create --name ten1epg1 --tenant tenant1 --port BL-1[0/11]
--switchport-mode trunk --ctag-range 1001 --l2-vni 1001:1001 --anycast-ip
1001:10.10.10.1/24
--vrf vrf1 --l3-vni 1001

efa tenant epg create --name ten2epg1 --tenant tenant2 --port BL-1[0/21]
--switchport-mode trunk --ctag-range 1101 --l2-vni 1101:1101 --anycast-ip
1101:10.10.11.1/24
--vrf vrf2 --l3-vni 1002
```

```
efa tenant create --name SharedTenant --port BL-1[0/1-8],BL-2[0/1-8] --type shared
```

```
efa tenant po create --name po101 --tenant SharedTenant --speed 10Gbps
--negotiation active --port BL-1[0/1],BL-1[0/2]

efa tenant po create --name po102 --tenant SharedTenant --speed 10Gbps
--negotiation active --port BL-1[0/3],BL-1[0/4]
```

VRF1

```
efa tenant epg create --name ten1epg2 --tenant tenant1 --type l3-handover
--po po101 --switchport-mode trunk --ctag-range 101 --vrf vrf1  --local-ipv4-address
11.1.1.1/30
--local-ipv6-address 2001:11:1:1::1/126 --remote-ipv4-address 11.1.1.2 --remote-ipv6-
address
2001:11:1:1::2 --remote-as 4220000001 --bfd --bfd-interval 100 --bfd-min-rx 200 --bfd-
multiplier 10
```

```
efa tenant epg create --name ten1epg3 --tenant tenant1 --type l3-handover
--po po102 --switchport-mode trunk --ctag-range 201 --vrf vrf1 --local-ipv4-address
12.1.1.1/30
--local-ipv6-address 2001:12:1:1::1/126 --remote-ipv4-address 12.1.1.2 --remote-ipv6-
address
2001:12:1:1::2 --remote-as 4220000001 --bfd --bfd-interval 100 --bfd-min-rx 200 --bfd-
multiplier 10
```

VRF2

```
efa tenant epg create --name ten2epg2 --tenant tenant2 --type l3-handover --po po101
--switchport-mode trunk --ctag-range 102 --vrf vrf2  --local-ipv4-address 11.2.1.1/30
--local-ipv6-address 2001:11:2:1::1/126 --remote-ipv4-address 11.2.1.2 --remote-ipv6-
address
2001:11:1:1::2 --remote-as 4220000001 --bfd --bfd-interval 100 --bfd-min-rx 200 --bfd-
multiplier 10
```

```
efa tenant epg create --name ten2epg3 --tenant tenant2 --type l3-handover --po po102
--switchport-mode trunk --ctag-range 202 --vrf vrf2 --local-ipv4-address 12.2.1.1/30
--local-ipv6-address 2001:12:2:1::1/126 --remote-ipv4-address 12.2.1.2 --remote-ipv6-
address
2001:12:2:1::2 --remote-as 4220000001 --bfd  --bfd-interval 100 --bfd-min-rx 200 --bfd-
multiplier 10
```

Centralized Routing

Routers can be configured either in centralized mode or distributed mode.

In a centralized router, the routing configurations are configured only on the border leaf pairs. In case of distributed mode, the routing configurations are configured on the corresponding leaf nodes where the endpoints reside.

During router creation, you can provide centralized mode or distributed mode as input.

- `openstack router create R1 --distributed`
- `openstack router create R2 --centralized`

The default option is centralized. Use the following command to create centralized routing:

`openstack router create R2`

The L3 service plug-in passes this information to EFA and the EFA Tenant Service creates VRF or routing configurations on border leafs or on leaf nodes based on the configuration mode.

Note

For the version 2.4 release, the OpenStack integration works with only a single pair of border leaf devices. A centralized routing instance is created on the single pair of border leaf devices. You must ensure that only a single border leaf pair is added as part of the fabric creation.

Preparing CLOS Fabric for Centralized Routing

Create a CLOS fabric containing border-leaf devices which can be used for the purpose of centralized routing.

```
efa fabric create --name <fabric-name> --type clos

efa fabric device add-bulk --name <fabric-name>
           --border-leaf <list-of-border-leaf-ip> --leaf <list-of-leaf-ip> --spine <list-
of-spine-ip>

efa fabric configure --name <fabric-name>
```

Preparing Non-CLOS Fabric for Centralized Routing

Create a Non-CLOS fabric containing border-leaf devices which can be used for the purpose of centralized routing.

```
efa fabric create --name <fabric-name> --type non-clos

efa fabric device add-bulk --name <fabric-name>
           --rack <leaf-rack-name> --ip <leaf-ip-pair>
           --border-leaf-rack <bl-rack-name> --border-leaf-ip <bl-ip-pair>

efa fabric configure --name <fabric-name>
```

Tenant VRF: Enable Centralized Routing

Users need to explicitly enable centralized routing at the tenant VRF level to override the default distributed routing behaviour.

Hence a given tenant can have multiple VRFs with some VRFs operating in distributed routing mode and some VRFs operating in centralized routing mode.

VRF Create

```
efa tenant vrf create --name <vrf-name> --tenant <tenant-name>
                      --routing-type {centralized | distributed}
                      --centralized-router <list-of-border-leaf-routers>
```

Example

```
efa tenant vrf create --name VRF1 --tenant tenant1
                  --routing-type centralized --centralized-router BL1-IP,BL2-IP
```

Tenant VRF: Provide physical routers for centralized routing

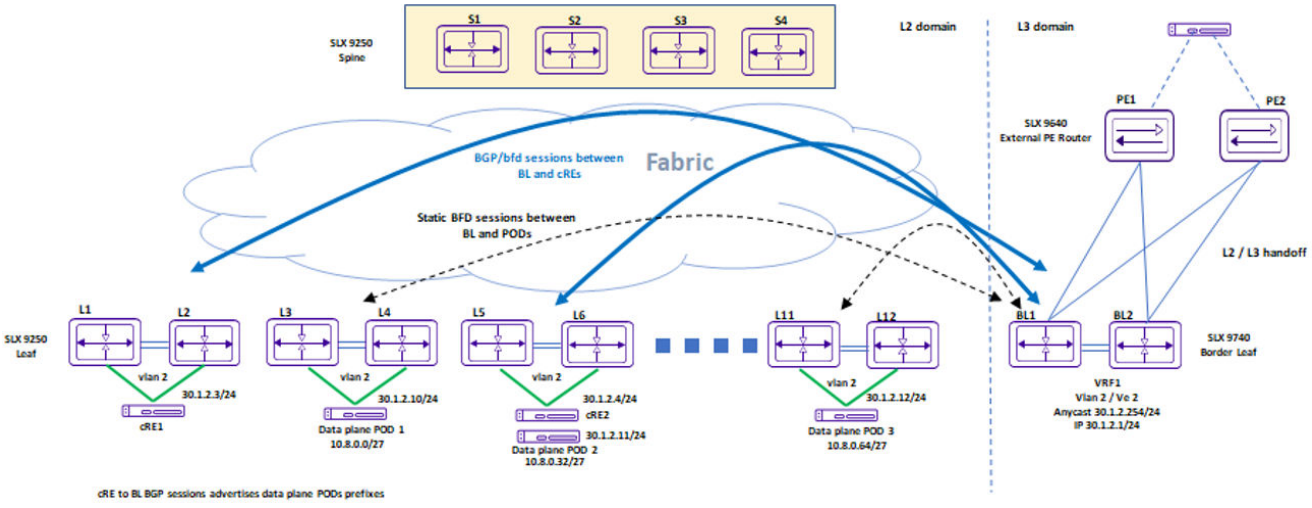
Provide a list of border-leaf IPs on which the centralized router (the VRF) needs to be instantiated upon. If a given fabric has only one BL pair and user hasn't provided any BL pair as centralized router, then the only available BL pair will be used as centralized router by default.

Provide only one BL pair on which the centailized router (the VRF) needs to be instantiated upon.

VRF instantiation happens on the border-leaf devices during the EPG create/update operations.

User will not be able to provide leaf/spine/super-spine ips as the target devices for centralized routing.

VRF (with centralized routing enabled) and its dependent L3 configuration (anycast-ip, local-ip, VRF static route, VRF static route bfd, router bgp static/dynamic peer, and router bgp peer-group) will be instantiated only on the border-leaf(s) on which the parent VRF exists.



When the centralized routing is enabled for a given tenant VRF:

1. Users must define the target border-leaf device on which the VRF needs to be instantiated.

   The VRF instantiation will happen only on those border-leaf devices and not on any other leaf/ border-leaf devices.
2. Users will not need to provide the target border-leaf device on which the anycast-ip needs to be configured.

   The anycast-ip configuration will happen *automatically* on the border-leaf devices on which the VRF is instantiated.
3. Users must provide the border-leaf ip (on which the VRF is instantiated) for the local-ip configuration.
4. Users must provide the border-leaf ip (on which the VRF is instantiated) for the VRF SR (Static Route) and VRF SR-BFD (Static Route – BFD) configuration.
5. Users must provide the border-leaf ip (on which the VRF is instantiated) for the BGP static and dynamic peer configuration.
6. Users must provide the border-leaf ip (on which the VRF is instantiated) for the BGP peer-group configuration.

VRF Create

```
efa tenant vrf create --name <vrf-name> --tenant <tenant-name>
                    --routing-type {centralized | distributed}
                       --centralized-router <list-of-border-leaf-routers>
```

Example

```
efa tenant vrf create --name VRF1 --tenant tenant1
                     --routing-type centralized --centralized-router BL1-IP,BL2-IP
```

Carving out of VRFs on the border-leaf pairs

Users must instantiate VRFs on the border-leaf devices based on the L3 scale requirements. Supposing the fabric has 100 VRFs with 4K anycast-ip, then user can instantiate all 100 VRFs on a single border-leaf pair. If the L3 scale requirements are higher than the scale supported by a single border-leaf pair then additional border-leaf pair needs to be added.

Tenant: EPG: anycast-ip

Anycast IP gets automatically configured on the border-leaf (BL1 and BL2) on which the VRF is instantiated.

```
efa tenant epg create --name ten1epg1 --tenant tenant1
                --port L1-IP[0/11],L2-IP[0/11]
                --switchport-mode trunk -ctag-range 11 --vrf VRF1 --anycast-ip
11:10.10.11.1/24
```

Tenant: EPG: local-ip

```
efa tenant epg create --name ten1epg1 --tenant tenant1 --vrf VRF1 --switchport-mode trunk
--ctag-range 11
                     --anycast-ip 11:10.10.11.1/24 --port L1-IP[0/1],L2-IP[0/1]
                   --local-ip 11,BL1-IP:11.22.33.41/24 --local-ip 11,BL2-IP:11.22.34.41/24
```

Tenant: VRF: static route

Users must provide the border-leaf ip (on which the VRF is instantiated) for the VRF SR (Static Route) and VRF SR-BFD (Static Route – BFD) configuration.

VRF Create

```
efa tenant vrf create --name <vrf-name> --tenant <tenant-name>
                   --ipv6-static-route-next-hop <border-leaf-ip, destination, next-hop,
distance>
                   --ipv4-static-route-next-hop <border-leaf-ip, destination, next-hop,
distance>
```

VRF Update

```
efa tenant vrf update -name <vrf-name> --tenant <tenant-name>
                   --operation <static-route-add|static-route-delete>
                   --ipv6-static-route-next-hop <border-leaf-ip, destination, next-hop,
distance>
                   --ipv4-static-route-next-hop <border-leaf-ip, destination, next-hop,
distance>
```

Example

```
efa tenant vrf create --name VRF1 --tenant tenant1
            --ipv6-static-route-next-hop BL1-IP,2000::/64,1001::2
            --ipv6-static-route-next-hop BL1-IP,2000::/64,1002::2
            --ipv6-static-route-next-hop BL2-IP,2001::/64,1001::2,4
            --ipv6-static-route-next-hop BL2-IP,2001::/64,1002::2
            --ipv4-static-route-next-hop BL1-IP,22.0.0.0/24,13.0.0.1,2
            --ipv4-static-route-next-hop BL1-IP,22.0.0.0/24,13.0.0.2
            --ipv4-static-route-next-hop BL2-IP,23.0.0.0/24,13.0.0.1
            --ipv4-static-route-next-hop BL2-IP,23.0.0.0/24,13.0.0.2
```

Tenant: VRF: static route bfd

VRF Create

```
efa tenant vrf create --name <vrf-name> --tenant <tenant-name>
                --ipv6-static-route-bfd <border-leaf-ip, destination-ip, source-ip, bfd-
min-tx, bfd-min-rx, bfd-multiplier>
                --ipv4-static-route-bfd <border-leaf-ip, destination-ip, source-ip, bfd-
min-tx, bfd-min-rx, bfd-multiplier>
```

VRF Update

```
efa tenant vrf update -name <vrf-name> --tenant <tenant-name>
                --operation <static-route-bfd-add|static-route-bfd-delete>
                --ipv6-static-route-bfd <border-leaf-ip, destination-ip, source-ip, bfd-
min-tx, bfd-min-rx, bfd-multiplier>
                --ipv4static-route-bfd  <border-leaf-ip, destination-ip, source-ip, bfd-
min-tx, bfd-min-rx, bfd-multiplier>
```

Example

```
efa tenant vrf create --name VRF1 --tenant tenant1
            --ipv6-static-route-bfd BL1-IP,1001::2,1001::1,100,200,5
            --ipv6-static-route-bfd BL2-IP,1011::2,1011::1,100,200,5
            --ipv6-static-route-bfd BL1-IP,1002::2, 1002::1
            --ipv6-static-route-bfd BL2-IP,1012::2, 1012::1
            --ipv4-static-route-bfd BL1-IP,13.0.0.1,13.0.0.9,200,300,6
            --ipv4-static-route-bfd BL2-IP,13.0.1.1,13.0.1.9,200,300,6
            --ipv4-static-route-bfd BL1-IP,13.0.0.2,13.0.0.10
            --ipv4-static-route-bfd BL2-IP,13.0.1.2,13.0.1.10
```

Tenan: BGP: peer-group

BGP peer-group create

```
efa tenant service bgp peer-group create --name <peer-group-name> --tenant <tenant-name>
      --description <description>
                        --pg-name <border-leaf-ip:pg-name> --pg-asn <border-leaf-ip:pg-
name,remote-asn>
                        --pg-bfd <border-leaf-ip:pg-name,bfd-enable(true/
false),interval,min-rx,multiplier>
                        --pg-next-hop-self <border-leaf-ip:pg-name,next-hop-self(true/
false)>
                        --pg-update-source-ip <border-leaf-ip:pg-name,update-source-ip>
```

Example

```
efa tenant service bgp peer-group create -name ten1BgpPG1 --tenant tenant1
                        --pg-name BL1-IP:pg1 --pg-asn BL1-
IP:pg1,6000
```

```
                          --pg-bfd BL1-IP:pg1,true,100,200,5
                          --pg-next-hop-self BL1-IP:pg1,true
                          --pg-update-source-ip BL1-IP:pg1,10.20.30.40
```

BGP peer-group update

```
efa tenant service bgp peer-group update --name <peer-group-name> --tenant <tenant-name>
                        --operation <peer-group-add|peer-group-delete|peer-group-desc-
update> --description <description>
                        --pg-name <border-leaf-ip:pg-name> --pg-asn <border-leaf-ip:pg-
name,remote-asn>
                        --pg-bfd <border-leaf-ip:pg-name,bfd-enable(true/
false),interval,min-rx,multiplier>
                        --pg-next-hop-self <border-leaf-ip:pg-name,next-hop-self(true/
false)>
                        --pg-update-source-ip <border-leaf-ip:pg-name,update-source-ip>
```

Example

```
efa tenant service bgp peer-group update --name ten1BgpPG1 --tenant tenant1
                --operation peer-group-add
                --pg-name BL1-IP:pg2 -pg-asn BL1-IP:pg2,7000
                --pg-bfd BL1-IP:pg2,true,200,300,6
                --pg-next-hop-self BL1-IP:pg2,true
                --pg-update-source-ip BL1-IP:pg2,10.20.30.41
```

Tenant: BGP: static peer

BGP static peer create

```
efa tenant service bgp peer create --name <peer-name> --tenant <tenant-
name>
          --ipv4-uc-nbr <border-leaf-ip,vrf-name:ipv4-neighbor,remote-
as>
          --ipv4-uc-nbr-bfd <border-leaf-ip,vrf-name:ipv4-neighbor,bfd-enable(true/
false),bfd-interval,bfd-rx,bfd-mult>
          --ipv4-uc-nbr-update-source-ip <border-leaf-ip,vrf-name:ipv4-neighbor,update-
source-ip>
          --ipv4-uc-nbr-next-hop-self <border-leaf-ip,vrf-name:ipv4-neighbor,next-hop-
self(true/false)>
          --ipv6-uc-nbr <border-leaf-ip,vrf-name:ipv6-neighbor,remote-as>
          --ipv6-uc-nbr-bfd <border-leaf-ip,vrf-name:ipv6-neighbor,bfd-enable(t/f),bfd-
interval,bfd-rx,bfd-mult>
          --ipv6-uc-nbr-update-source-ip <border-leaf-ip,vrf-name:ipv6-neighbor,update-
source-ip>
          --ipv6-uc-nbr-next-hop-self <border-leaf-ip,vrf-name:ipv6-neighbor,next-hop-
self(true/false)>
```

Example:

```
efa tenant service bgp peer create --name bgpservice1 --tenant
tenant1
            --ipv4-uc-nbr BL1-IP,VRF1:10.20.30.40,5000
            --ipv4-uc-nbr-bfd BL1-IP,VRF1:10.20.30.40,true,100,200,5
            --ipv4-uc-nbr-update-source-ip BL1-IP,VRF1:10.20.30.40,11.22.20.33
            --ipv4-uc-nbr-next-hop-self BL1-IP,VRF1:10.20.30.40,true
```

BGP static peer update

```
efa tenant service bgp peer update --name <peer-name> --tenant <tenant-name>
             --operation peer-add
             --ipv4-uc-nbr <border-leaf-ip,vrf-name:ipv4-neighbor,remote-as>
             --ipv4-uc-nbr-bfd <border-leaf-ip,vrf-name:ipv4-neighbor,bfd-enable(t/f),bfd-
```

```
interval,bfd-rx,bfd-mult>
             --ipv4-uc-nbr-update-source-ip <border-leaf-ip,vrf-name:ipv4-neighbor,update-
source-ip>
             --ipv4-uc-nbr-next-hop-self <border-leaf-ip,vrf-name:ipv4-neighbor,next-hop-
self(true/false)>
             --ipv6-uc-nbr <border-leaf-ip,vrf-name:ipv6-neighbor,remote-as>
             --ipv6-uc-nbr-bfd <border-leaf-ip,vrf-name:ipv6-neighbor,bfd-enable(t/f),bfd-
interval,bfd-rx,bfd-mult>
             --ipv6-uc-nbr-update-source-ip <border-leaf-ip,vrf-name:ipv6-neighbor,update-
source-ip>
             --ipv6-uc-nbr-next-hop-self <border-leaf-ip,vrf-name:ipv6-neighbor,next-hop-
self(true/false)>
```

Example:

```
efa tenant service bgp peer update --name bgpservice1 --tenant tenant1 --operation peer-
add
            --ipv6-uc-nbr BL1-IP,VRF1:10::40,5000
            --ipv6-uc-nbr-bfd BL1-IP,VRF1:10::40,true,100,200,5
            --ipv6-uc-nbr-update-source-ip BL1-IP,VRF1:10::40,11::22
            --ipv6-uc-nbr-next-hop-self BL1-IP,VRF1:10::40,true
```

Tenant: BGP: Dynamic peer

BGP dynamic peer create

```
efa tenant service bgp peer create --name <peer-name> --tenant <tenant-name>
                               --ipv4-uc-dyn-nbr <border-leaf-ip,vrf-name:listen-
range,peer-group-name,listen-limit>
                               --ipv6-uc-dyn-nbr <border-leaf-ip,vrf-name:listen-
range,peer-group-name,listen-limit>
```

Example:

```
efa tenant service bgp peer create --name bgpservice1 --tenant tenant1
                    --ipv4-uc-dyn-nbr BL1-IP,VRF1:11.22.33.44/30,pg1,10
```

BGP dynamic peer update

```
efa tenant service bgp peer update --name <peer-name> --tenant <tenant-name>
                                --operation peer-add
                                --ipv4-uc-dyn-nbr <border-leaf-ip,vrf-name:listen-
range,peer-group-name,listen-limit>
                                --ipv6-uc-dyn-nbr <border-leaf-ip,vrf-name:listen-
range,peer-group-name,listen-limit>
```

Example:

```
efa tenant service bgp peer create --name bgpservice1 --tenant
tenant1
                          --operation peer-add –ipv4-uc-dyn-nbr BL1-
IP,VRF1:11::22/127,pg1,20
```

Administered Partial Success

For a two-leaf MCT pair, you can enable configurations to succeed when only one node of the pair is reachable.

Overview

By default, when a REST operation succeeds on one device but fails on another, configuration changes are rolled back for both devices. For more information, see Rollback Scenarios for Data Consistency on page 42.

However, for a two-leaf MCT pair, you can administratively change the process to permit configuration to succeed even when one device is down. This process, called an administered partial success, is as follows.

- You use the **`efa inventory admin-state`** command to change the state of the unreachable device to "admin down." The device then goes into maintenance mode. For more information about changing a device state, see Administratively Manage a Device State on page 127.
- EFA filters out configurations destined for MCT pair as follows.
  - Create-related and delete-related configurations destined for the "admin up" device succeed.
  - Create-related configurations are not attempted for the "admin down" device, but the configurations are considered a success. These configurations are marked as pending, to be pushed to the device when it comes back up.
  - Delete-related configurations (de-configurations) are not attempted for the "admin down" device and the operation fails with an error in the REST response. You can retry these de-configurations after the device transitions to “admin up” state.

    The reason being EFA doesn't want to leave stale configurations on the devices. If stale configurations are left on the devices, then bringing the devices (with stale configurations) back into EFA will be erroneous considering the full brownfield support is missing in EFA.
- When the device is again reachable, you change the state of the device to "admin-up."
- EFA pushes the pending configurations to the device, and the drift and reconcile process ensures that the configurations in EFA and the device are synchronized. For more information, see Drift and Reconcile on page 39.
- The device comes out of maintenance mode.

Tips and considerations

- You can use Switch Health Management to verify the reachability of a device. Use the **`--health-check-interval`** and **`--health-check-heartbeat-miss-threshold`** settings of the **`efa inventory device setting update`** command. For more information, see Monitor Switch Health on page 160.
- You can retry the same CLI or REST operation after the “admin down” devices transition to “admin up” state so that the deconfiguration will be attempted on all the devices. You can the “force” option available in the REST API to forcefully delete the entities from EFA even in case of partial success topology.

- You can use the **efa tenant debug device drift** command to determine any drift between the intended EFA configuration and the device configuration. These commands also identify the app state and the dev state: **efa tenant epg show** and **efa tenant po show**.
- EFA will block the tenant reconciliation API and rest of the tenant APIs will support partial success behaviour.
- If a high-availability failover or restart occurs while a device is in "admin down" or "admin up" state, you must reapply the state.
- If an operation such as drift and reconcile or a firmware download is in progress when you submit the command to change the state, the command is blocked until the operation is complete.
- This feature is supported only for devices in an MCT pair. Standalone devices are not supported.
- You can change the status of only one device in an MCT pair to "admin down" to benefit from administered partial success.
  - When both devices are in "admin down" state, the topology is considered a complete failure. Configuration attempts on these devices are rejected and error messages are returned in the REST responses. Administered partial success is not applicable.
  - When both devices are in "admin up" state, the topology is considered a complete success. Configuration attempts on these devices are accepted. Administered partial success is not applicable.

Behavior changes during "admin down" state

After a device state changes to "admin down," the following behavior changes occur.

- Switch Health Management does not trigger the drift and reconcile process.
- A device going into maintenance mode does not trigger the drift and reconcile process.
- The following commands are blocked from affecting the device.

Table 14: Blocked commands

Command type and name
Inventory commands
efa inventory device compare --ip
efa inventory drift-reconcile --ip
efa inventory device setting update --ip
efa inventory rma --ip
efa inventory config-backup execute --ip
efa inventory config-replay execute --ip
efa inventory device update --fabric
efa inventory device firmware-download prepare add --ip
efa inventory device update --ip
efa inventory device interface set-speed --ip
efa inventory device interface set-breakout --ip
efa inventory device interface unset-breakout --ip

Table 14: Blocked commands (continued)

Command type and name
efa inventory device interface set-mtu --ip
efa inventory device system set-mtu --ip
efa inventory device interface set-admin-state --ip
efa inventory device running-config persist --ip
Fabric commands
efa fabric configure --name
efa fabric device remove --ip <> --name <> Allowed with the --no-device-cleanup option.
efa fabric show-config --name
efa fabric topology show underlay --name
efa fabric topology show overlay --name
efa fabric topology show physical --name

Behavior changes during "admin up" state

After a device is returned to "admin up" state and after the drift and reconcile process is complete (which the state change triggers), Switch Health Management and drift and reconcile resume normal behavior. Also, the blocked commands are unblocked.

Administratively Manage a Device State

You can administratively manage the state of an SLX device using the EFA command line.

You can change a state to up or down, delete a state from the history, and view the state history and the current state. For details about the command and its parameters, see the Extreme Fabric Automation Command Reference, 2.4.0.

1. To change a device to the up state, run the following command.

```
$ efa inventory admin-state up --ip <device IP>
AdminStateUp [success]
Admin State Up execution UUID: 8d9fa0cf-dc76-42cc-ac7a-57902a47c1b2
```

   This example changes the state for the specified IP address and generates a UUID, which you can use in the **`efa inventory admin-state detail`** version of the command.

2. To change a device to the down state, run the following command.

```
$ efa inventory admin-state down --ip <device IP>
AdminStateDown [success]
Admin State Down execution UUID: 28eb0845-7a7a-4851-b453-b3020c6900f2
```

   This example changes the state for the specified IP address and generates a UUID, which you can use in the **`efa inventory admin-state detail`** version of the command.

3. To review the details of a state change, run the following command.

```
$ efa inventory admin-state detail --uuid 28eb0845-7a7a-4851-b453-b3020c6900f2
```

4. To view the history of the admin changes for a specified device, run the following command.

```
# efa inventory admin-state history --ip <device IP>
```

5. To display the admin state and the health check state of a device, run the following command.

```
$ efa inventory admin-state show --ip <device IP>
```

6. To delete the instance of the admin state change for a device, run the following command.

```
$ efa inventory admin-state delete --key <device IP or UUID>
```

APS Behaviour of Tenant Configuration

Existing behaviour in EFA 2.3.0

Configuration or Deconfiguration is never attempted on admin down switching devices.

Target Devices

Devices on which the configuration is intended to be pushed.

Complete Failure Topology

- Topology having atleast one single-homed device in admin down state or atleast one dual-homed device pair with both the devices in admin down state is a “complete failure topology”.
- Any Tenant CLI/REST of create or delete nature attempted on the target devices having “complete failure topology” is rejected with an appropriate error and result in a complete failure. EFA does not not have any configuration recipe prepared for this REST API or CLI as the entire request is rejected. For example, an EPG create attempted on a single-homed device which is admin down state.

Complete Success Topology

- Topology having all the single-homed devices and all the dual-homed device pair in admin up state is a “complete success topology”.
- Any Tenant CLI/REST of create/delete nature attempted on the target devices having “complete success topology” will result in the configuration recipe preparation for all the target devices and configuration will be attempted on all the target devices as all the target devices are in “admin up” state.

Partial Success Topology

- Topology which is not “complete failure topology” and having at least one dual-homed device pair with one of the device in admin down state and the other device in admin up state is a “partial success topology”.
- Any Tenant CLI/REST of create nature attempted on the target devices having “partial success topology” will result in the configuration being attempted onto the “admin up” devices and configuration not being attempted onto the “admin down” devices. Even though configuration is not attempted for “admin down” devices, the configuration will be treated as success for the “admin down” devices.

  Configuration recipe will be prepared and persisted in EFA for all the devices and the configuration will be auto reconciled with the devices when the “admin down” devices transition to “admin up”.

  Hence when the devices are “admin up” both EFA intended configuration and device configuration will be in sync.
- Any Tenant CLI/REST of delete nature attempted on the target devices having “partial success topology” will result in deconfiguration attempted on the “admin up” devices and deconfiguration not attempted on the “admin down” devices and the CLI/REST operation will fail with an appropriate error indicating that the deconfiguration not being attempted on the “admin down” devices.

  The reason being EFA doesnt want to leave stale configurations on the devices. If stale configurations are left on the devices, then bringing the devices (having stale configurations) back into EFA will be erroneous considering the full brownfield support is missing in EFA. User can retry the same CLI/REST operation after the “admin down” devices transition to “admin up” state so that the deconfiguration will be attempted on all the devices. User can always use “force” option available in REST API to forcefully delete the entities from EFA even in case of partial success topology.
- Drift between the EFA intended configuration and device config will be shown in the “efa tenant debug device drift” CLI/REST output as well as in the corresponding entity GET/SHOW output i.e. “efa tenant epg show”, “efa tenant po show” etc. in the form of “app-state” and “dev-state”.
- EFA will block the tenant reconciliation API and rest of the tenant APIs will support partial success behaviour.

APS: Pre Provisioning Support by modifying the target device list to include the MCT neighbour

Scenario	2.3.2 Target Device List	2.4.0 Target Device List	EFA 2.3.2 Resultant Topology	EFA 2.4.0 Resultant Topology
Single Homed PO Create with PO member on Rack1Device2	Rack1Device2	Rack1Device1 Rack1Device2	Complete Failure	Partial Success
EPG create with CEP member on Rack1Device2	Rack1Device2 Rack1Device1	Rack1Device1 Rack1Device2	Partial Success	Partial Success
BGP Peer Group create with the peer-group residing on Rack1Device2	Rack1Device2	Rack1Device1 Rack1Device2	Complete Failure	Partial Success
BGP Peer Create with the static/dynamic BGP peers residing on Rack1Device2	Rack1Device2	Rack1Device1 Rack1Device2	Complete Failure	Partial Success
VRF update with SR/SR-BFD residing on Rack1Device2	Rack1Device2	Rack1Device1 Rack1Device2	Complete Failure	Partial Success

```
efa tenant show
+------+-------------+-------------+------------+-----------+-----------+---------
+----------------------+
| Name | L2VNI-Range | L3VNI-Range | VLAN-Range | VRF-Count | Enable-BD |  Type
|        Ports         |
+------+-------------+-------------+------------+-----------+-----------+---------
+----------------------+
| ten1 |             |             | 11-20      | 10        | False     | private |
10.20.246.15[0/1-10] |
|      |             |             |            |           |           |         |
10.20.246.16[0/1-10] |
+------+-------------+-------------+------------+-----------+-----------+---------
+----------------------+
Tenant Details

efa inventory admin-state down --ip 10.20.246.15
AdminStateDown [success]
Admin State Down execution UUID: 6eaa1ebe-40fe-4628-8d5c-df11ffc4521e
execute the CLI to get details : efa inventory admin-state detail --uuid
6eaa1ebe-40fe-4628-8d5c-df11ffc4521e

efa inventory admin-state detail --uuid 6eaa1ebe-40fe-4628-8d5c-df11ffc4521e
+--------------------------------+----------------------------------------+
|              NAME              |                VALUE                   |
+--------------------------------+----------------------------------------+
| UUID                           | 6eaa1ebe-40fe-4628-8d5c-df11ffc4521e |
+--------------------------------+----------------------------------------+
| Device IP                      | 10.20.246.15                           |
+--------------------------------+----------------------------------------+
| Admin State Action             | down                                   |
+--------------------------------+----------------------------------------+
```

```
| Status                          | success                                |
+---------------------------------+----------------------------------------+
| Fabric Status                   | success                                |
+---------------------------------+----------------------------------------+
| Tenant Status                   | success                                |
+---------------------------------+----------------------------------------+
| Maintenance Mode Enable Status | success                                 |
+---------------------------------+----------------------------------------+
| Start Time                      | 2021-02-06 21:18:53 -0800 PST          |
+---------------------------------+----------------------------------------+
| Last Modified                   | 2021-02-06 21:19:59 -0800 PST          |
+---------------------------------+----------------------------------------+
| Duration                        | 1m5.517263907s                         |
+---------------------------------+----------------------------------------+
```

Behaviour in 2.3.2

```
efa tenant po create --name ten1po1 --tenant ten1 --port 10.20.246.15[0/1-2] --speed
10Gbps --negotiation active
    PortChannel creation failed:
            Error: Device 10.20.246.15 is administratively down

efa tenant po create --name ten1po1 --tenant ten1 --port 10.20.246.16[0/1-2] --speed
10Gbps --negotiation active
    PortChannel created successfully.

efa tenant epg create --name ten1epg1 --tenant ten1 --po ten1po1 --switchport-mode trunk
--ctag-range 11-12 --anycast-ip 11:10.0.11.1/24 --anycast-ip 12:10.0.12.1/24 --vrf
ten1vrf1 --l2-vni 11:11 --l2-vni 12:12 --l3-vni 8192
    EndpointGroup created successfully.

efa tenant service bgp peer create --name ten1bgppeer1 --tenant ten1 --ipv4-uc-nbr
10.20.246.15,ten1vrf1:10.0.0.0,65001
    BgpService creation Failed:
            Error: Devices [10.20.246.15] are administratively down

efa tenant service bgp peer-group create --name ten1bgppeergroup1 --tenant ten1 --pg-name
10.20.246.15:pg1 --pg-asn 10.20.246.15,pg1:65010
    BgpService creation Failed:
            Error: Devices [10.20.246.15] are administratively down
```

Behaviour in 2.4.0

```
efa tenant po create --name ten1po1 --tenant ten1 --port 10.20.246.15[0/1-2] --speed
10Gbps --negotiation active

efa tenant epg create --name ten1epg1 --tenant ten1 --po ten1po1 --switchport-mode trunk
--ctag-range 11-12 --anycast-ip 11:10.0.11.1/24 --anycast-ip 12:10.0.12.1/24 --vrf
ten1vrf1

efa tenant service bgp peer create --name ten1bgppeer1 --tenant ten1 --ipv4-uc-nbr
10.20.246.15,ten1vrf1:10.0.0.0,65001

efa tenant service bgp peer-group create --name ten1bgppeergroup1 --tenant ten1 --pg-name
10.20.246.15:pg1 --pg-asn 10.20.246.15,pg1:65010

efa tenant po show --name ten1po1 --tenant ten1
+---------+--------+----+--------+-------------+----------+---------+---------------------
+------------+-----------------+-----------+
|  Name   | Tenant | ID | Speed  | Negotiation | Min Link |  Lacp   |        Ports
|   State    |    Dev State    | App State |
|         |        |    |        |             |  Count   | Timeout |
|            |                 |           |
+---------+--------+----+--------+-------------+----------+---------+---------------------
```

```
+-----------+----------------+-----------+
| ten1po1 |  ten1  | 1  | 10Gbps |   active    |    1     |  long   | 10.20.246.15[0/1-2]
| po-created | not-provisioned | cfg-ready |
+---------+--------+----+--------+-------------+----------+---------+--------------------
+-----------+----------------+-----------+

efa tenant vrf show --name ten1vrf1 --tenant ten1
(efa:root)root@node-2:~# efa tenant vrf show --name ten1vrf1 --tenant ten1
+----------+--------+-------------+--------------------+-----------------+----------
+-----------+-----------+------------+-----------------+-----------+
|   Name   | Tenant | Routing Type | Centralized Routers |   Redistribute   | Max Path |
Local Asn | Enable GR |   State    |    Dev State    | App State |
+----------+--------+-------------+--------------------+-----------------+----------
+-----------+-----------+------------+-----------------+-----------+
| ten1vrf1 |  ten1  | distributed  |                    | connected,static |    50
|   65002   |   false   | vrf-create | not-provisioned | cfg-ready |
+----------+--------+-------------+--------------------+-----------------+----------
+-----------+-----------+------------+-----------------+-----------+


efa tenant epg show --detail
================================================================================================
=====================================================================
Name          : ten1epg1
Tenant        : ten1
Description   :
Type          : extension
Ports         :
POs           :
              : unstable            : ten1po1
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 11-12
              : vrf                 : ten1vrf1 [unstable]
              : l3-vni              : 8192
Network Property [Flags : * - Native Vlan]
+------+--------+-------------+-------------+---------+-------------------------------
+-------------------------+-------------+-----------------------+---------------------
+-----------------+-----------+
| Ctag | L2-Vni | Anycast-IPv4 | Anycast-IPv6 | BD-name | Local IP (Device-IP->Local-IP)
|    Ctag-Description     | Mtu-IPv6-ND | ManagedConfig-IPv6-ND | OtherConfig-IPv6-ND
|    Dev-state    | App-state |
+------+--------+-------------+-------------+---------+-------------------------------
+-------------------------+-------------+-----------------------+---------------------
+-----------------+-----------+
| 11   | 11     | 10.0.11.1/24 |             |         |
| Tenant L3 Extended VLAN |             | False                 | False               |
not-provisioned | cfg-ready |
+------+--------+-------------+-------------+---------+-------------------------------
+-------------------------+-------------+-----------------------+---------------------
+-----------------+-----------+
| 12   | 12     | 10.0.12.1/24 |             |         |
| Tenant L3 Extended VLAN |             | False                 | False               |
not-provisioned | cfg-ready |
+------+--------+-------------+-------------+---------+-------------------------------
+-------------------------+-------------+-----------------------+---------------------
+-----------------+-----------+

efa tenant service bgp peer-group show
================================================================================================
=====================================================================
Name        : ten1bgppeergroup1
Tenant      : ten1
State       : bgp-pg-state-created
```

```
Description :
+--------------+-----------------+------------+-------------+--------------+--------
+----------------+----------------+-----------------+-----------------+-----------+
|  Device IP   | PEER-GROUP-NAME | REMOTE ASN | BFD Enabled | BFD Interval | BFD Rx | BFD
Multiplier | Next-Hop-Self | Update-Source-IP |    Dev-state    | App-state |
+--------------+-----------------+------------+-------------+--------------+--------
+----------------+----------------+-----------------+-----------------+-----------+
| 10.20.246.15 | pg1             | 65010      | false       | 0            | 0      |
0              | false         |                 | not-provisioned | cfg-ready |
+--------------+-----------------+------------+-------------+--------------+--------
+----------------+----------------+-----------------+-----------------+-----------+

efa tenant service bgp peer show
================================================================================================
================================================================================================
========
Name        : ten1bgppeer1
Tenant      : ten1
State       : bs-state-created
Description :
Static Peer:
+--------------+----------+------+---------+-----------+------------+-------------
+--------------+--------+---------------+---------------+------------------
+-----------------+-----------+
|  Device IP   |   VRF    | AFI  |  SAFI   | REMOTE IP | REMOTE ASN | BFD Enabled | BFD
Interval | BFD Rx | BFD Multiplier | Next Hop Self | Update Source IP |    Dev-state    |
App-state |
+--------------+----------+------+---------+-----------+------------+-------------
+--------------+--------+---------------+---------------+------------------
+-----------------+-----------+
| 10.20.246.15 | ten1vrf1 | ipv4 | unicast | 10.0.0.0  | 65001      | false       |
0            | 0      | 0              | false         |                  | not-
provisioned | cfg-ready |
+--------------+----------+------+---------+-----------+------------+-------------
+--------------+--------+---------------+---------------+------------------
+-----------------+-----------+
Dynamic Peer:
+-----------+-----+-----+------+--------------+------------+--------------+-----------
+-----------+
| Device IP | VRF | AFI | SAFI | Listen Range | Peer Group | Listen Limit | Dev-state |
App-state |
+-----------+-----+-----+------+--------------+------------+--------------+-----------
+-----------+
================================================================================================
================================================================================================
========

efa inventory admin-state up --ip 10.20.246.15
AdminStateUp [success]
Admin State Up started execution UUID: 0ae0db00-c0de-4d55-8410-6d67bca4ad65


efa inventory admin-state detail --uuid 0ae0db00-c0de-4d55-8410-6d67bca4ad65
+------------------------------+----------------------------------------+
|            NAME              |                VALUE                   |
+------------------------------+----------------------------------------+
| UUID                         | 0ae0db00-c0de-4d55-8410-6d67bca4ad65 |
+------------------------------+----------------------------------------+
| Device IP                    | 10.20.246.15                           |
+------------------------------+----------------------------------------+
| Admin State Action           | up                                     |
+------------------------------+----------------------------------------+
| Status                       | success                                |
+------------------------------+----------------------------------------+
```

```
| Fabric Status               | success                                |
+-----------------------------+----------------------------------------+
| Tenant Status               | success                                |
+-----------------------------+----------------------------------------+
| Drift and Reconcile id      | a3c987ea-f709-4f7c-8ae4-bc5c9481cc55 |
+-----------------------------+----------------------------------------+
| Drift and Reconcile Status  | DR Completed                           |
+-----------------------------+----------------------------------------+
| Start Time                  | 2021-02-06 21:39:09 -0800 PST        |
+-----------------------------+----------------------------------------+
| Last Modified               | 2021-02-06 21:47:09 -0800 PST        |
+-----------------------------+----------------------------------------+
| Duration                    | 8m0.126957723s                         |
+-----------------------------+----------------------------------------+
--- Time Elapsed: 47.334754ms ---
(efa:root)root@node-2:~#

efa tenant vrf show --name ten1vrf1 --tenant ten1
+----------+--------+-------------+--------------------+-----------------+----------
+-----------+-----------+--------------------+-------------+-------------+
|   Name   | Tenant | Routing Type | Centralized Routers |   Redistribute   | Max Path |
Local Asn | Enable GR |       State        |  Dev State  |  App State  |
+----------+--------+-------------+--------------------+-----------------+----------
+-----------+-----------+--------------------+-------------+-------------+
| ten1vrf1 |  ten1  | distributed  |                    | connected,static |    50
|   65002   |   false   | vrf-device-created | provisioned | cfg-in-sync |
+----------+--------+-------------+--------------------+-----------------+----------
+-----------+-----------+--------------------+-------------+-------------+

efa tenant po show
+---------+--------+----+-------------------------+--------+-------------+--------------
+---------------------+------------+------------+-------------+--------------+
|  Name   | Tenant | ID |       Description        | Speed  | Negotiation | MinLinkCount
|        Ports        | LacpTimeout |   State    |  Dev-State  |   App-State  |
+---------+--------+----+-------------------------+--------+-------------+--------------
+---------------------+------------+------------+-------------+--------------+
| ten1po1 | ten1   | 1  | EFA Port-channel ten1po1 | 10Gbps | active      | 1
| 10.20.246.15[0/1-2] | long        | po-created | provisioned |  cfg-in-sync |
+---------+--------+----+-------------------------+--------+-------------+--------------
+---------------------+------------+------------+-------------+--------------+

efa tenant epg show
================================================================================================
=====================================================================
Name          : ten1epg1
Tenant        : ten1
Description   :
Type          : extension
Ports         :
POs           :
              : unstable            : ten1po1
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 11-12
              : vrf                 : ten1vrf1
              : l3-vni              : 8192
Network Property [Flags : * - Native Vlan]
+------+--------+--------------+--------------+---------+-------------------------------
+-------------------------+-------------+----------------------+---------------------
+-------------+--------------+
| Ctag | L2-Vni | Anycast-IPv4 | Anycast-IPv6 | BD-name | Local IP (Device-IP->Local-IP)
|    Ctag-Description     | Mtu-IPv6-ND | ManagedConfig-IPv6-ND | OtherConfig-IPv6-ND |
Dev-state  |   App-state  |
+------+--------+--------------+--------------+---------+-------------------------------
```

```
+-------------------------+-------------+----------------------+--------------------
+-------------+--------------+
| 11   | 11     | 10.0.11.1/24 |              |         |
| Tenant L3 Extended VLAN |              | False                | False               |
provisioned |  cfg-in-sync |
+------+--------+--------------+--------------+---------+--------------------------------
+-------------------------+-------------+----------------------+--------------------
+-------------+--------------+
| 12   | 12     | 10.0.12.1/24 |              |         |
| Tenant L3 Extended VLAN |              | False                | False               |
provisioned |  cfg-in-sync |
+------+--------+--------------+--------------+---------+--------------------------------
+-------------------------+-------------+----------------------+--------------------
+-------------+--------------+
For 'unstable' entities, run 'efa tenant po/vrf show' for details
================================================================================================
=====================================================================


efa tenant service bgp peer-group show
================================================================================================
=====================================================================
Name        : ten1bgppeergroup1
Tenant      : ten1
State       : bgp-pg-state-created
Description :
+--------------+-----------------+------------+-------------+--------------+--------
+----------------+---------------+------------------+-------------+-------------+
|  Device IP   | PEER-GROUP-NAME | REMOTE ASN | BFD Enabled | BFD Interval | BFD Rx | BFD
Multiplier | Next-Hop-Self | Update-Source-IP |  Dev-state  |  App-state  |
+--------------+-----------------+------------+-------------+--------------+--------
+----------------+---------------+------------------+-------------+-------------+
| 10.20.246.15 | pg1             | 65010      | false       | 0            | 0      |
0              | false         |                  | provisioned | cfg-in-sync |
+--------------+-----------------+------------+-------------+--------------+--------
+----------------+---------------+------------------+-------------+-------------+
================================================================================================
=====================================================================

efa tenant service bgp peer show
================================================================================================
================================================================================================
========
Name        : ten1bgppeer1
Tenant      : ten1
State       : bs-state-created
Description :
Static Peer:
+--------------+----------+------+---------+-----------+------------+-------------
+--------------+--------+---------------+---------------+------------------+-------------
+-------------+
|  Device IP   |   VRF    | AFI  |  SAFI   | REMOTE IP | REMOTE ASN | BFD Enabled | BFD
Interval | BFD Rx | BFD Multiplier | Next Hop Self | Update Source IP |  Dev-state  |
App-state  |
+--------------+----------+------+---------+-----------+------------+-------------
+--------------+--------+---------------+---------------+------------------+-------------
+-------------+
| 10.20.246.15 | ten1vrf1 | ipv4 | unicast | 10.0.0.0  | 65001      | false       |
0            | 0      | 0              | false         |                  | provisioned |
cfg-in-sync |
+--------------+----------+------+---------+-----------+------------+-------------
+--------------+--------+---------------+---------------+------------------+-------------
+-------------+
Dynamic Peer:
```

```
+-----------+-----+-----+------+-------------+-----------+-------------+-----------
+-----------+
| Device IP | VRF | AFI | SAFI | Listen Range | Peer Group | Listen Limit | Dev-state |
App-state |
+-----------+-----+-----+------+-------------+-----------+-------------+-----------
+-----------+
================================================================================
================================================================================
========
```

APS: Deletion support for the pre-provisioned config

Problem in EFA 2.3.2

Creation of EFA entities (PO/EPG etc.) on an admin down device followed by the Deletion of the same EFA entities (PO/EPG etc) on the same admin down device used to fail even though the configuration was never pushed to the devices.

Solution in EFA 2.4.0

Succeed the deletion of the EFA entities (PO/EPG etc.) if the resultant configuration to be deleted has never been pushed to the devices.

Pre-provisioned config

The pre-provisioned config is present in EFA DB and not present on the SLX.

```
efa inventory admin-state  show --
ip 10.20.246.15
+----------------------
+--------------+
|          NAME          |
VALUE     |
+----------------------
+--------------+
| Device IP             |
10.20.246.15 |
+----------------------
+--------------+
| Admin State           |
down          |
+----------------------
+--------------+
| Health Check Status   |
Disable       |
+----------------------
+--------------+
```

```
efa inventory admin-state  show --
ip 10.20.246.16
+----------------------
+--------------+
|          NAME          |
VALUE     |
+----------------------
+--------------+
| Device IP             |
10.20.246.16 |
+----------------------
+--------------+
| Admin State           |
up            |
+----------------------
+--------------+
| Health Check Status   |
Disable       |
+----------------------
+--------------+
```

```
efa tenant po create --name ten1po1 --tenant ten1 --port
10.20.246.15[0/1],10.20.246.16[0/1] --speed 10Gbps --negotiation active
efa tenant vrf create --name ten1vrf1 --tenant ten1
efa tenant epg create --name ten1epg1 --tenant ten1 --po ten1po1 --switchport-mode trunk
--ctag-range 11-12 --anycast-ip 11:10.0.11.1/24 --anycast-ip 12:10.0.12.1/24 --vrf
ten1vrf1
efa tenant service bgp peer create --name ten1bgppeer1 --tenant ten1 --ipv4-uc-nbr
10.20.246.15,ten1vrf1:10.0.0.0,65001 --ipv4-uc-nbr 10.20.246.16,ten1vrf1:10.1.0.0,65001
efa tenant service bgp peer-group create --name ten1bgppeergroup1 --tenant ten1 --pg-name
10.20.246.15:pg1 --pg-asn 10.20.246.15,pg1:65010  --pg-name 10.20.246.16:pg1 --pg-asn
10.20.246.16,pg1:65010
```

```
efa tenant po show
+---------+--------+----+--------------------------+--------+-------------+--------------
+--------------------+-------------+------------+-----------------+-----------+
|  Name   | Tenant | ID |       Description        | Speed  | Negotiation | MinLinkCount
|       Ports        | LacpTimeout |   State    |    Dev-State    | App-State |
+---------+--------+----+--------------------------+--------+-------------+--------------
+--------------------+-------------+------------+-----------------+-----------+
| ten1po1 | ten1   | 1  | EFA Port-channel ten1po1 | 10Gbps | active      | 1
| 10.20.246.15[0/1] | long        | po-created | not-provisioned | cfg-ready |
|         |        |    |                          |        |             |
| 10.20.246.16[0/1] |             |            |                 |           |
+---------+--------+----+--------------------------+--------+-------------+--------------
+--------------------+-------------+------------+-----------------+-----------+

efa tenant service bgp peer-group show
================================================================================================
=====================================================================
Name        : ten1bgppeergroup1
Tenant      : ten1
State       : bgp-pg-state-created
Description :
+--------------+-----------------+------------+-------------+--------------+--------
+----------------+---------------+-----------------+-----------------+-------------+
|  Device IP   | PEER-GROUP-NAME | REMOTE ASN | BFD Enabled | BFD Interval | BFD Rx | BFD
Multiplier | Next-Hop-Self | Update-Source-IP |    Dev-state    |  App-state  |
+--------------+-----------------+------------+-------------+--------------+--------
+----------------+---------------+-----------------+-----------------+-------------+
| 10.20.246.16 | pg1             | 65010      | false       | 0            | 0      |
0              | false         |                 | provisioned     | cfg-in-sync |
+--------------+-----------------+------------+-------------+--------------+--------
+----------------+---------------+-----------------+-----------------+-------------+
| 10.20.246.15 | pg1             | 65010      | false       | 0            | 0      |
0              | false         |                 | not-provisioned | cfg-ready   |
+--------------+-----------------+------------+-------------+--------------+--------
+----------------+---------------+-----------------+-----------------+-------------+
================================================================================================
=====================================================================

efa tenant vrf show --name ten1vrf1 --tenant ten1
================================================================================================
=====================================================================
Name                    : ten1vrf1
Vrf State               : vrf-device-created
Vrf Device State        : not-provisioned
Vrf App State           : cfg-ready
Tenant Name             : ten1
Routing Type            : distributed
L3 VNI                  : 8191
IRB BD                  : 4095
IRB VE                  : 8191
BR BD                   :
BR VE                   :
BR VNI                  : 4096
RH max path             :
RH ecmp enable          :
Graceful restart enable :
Route Target            : import 101:101
                        : export 101:101
Static Route            : Switch-IP->Network,Nexthop-IP[Route-Distance], ...
                        :
Local Asn               :
Static Route BFD        : Switch-IP->[DestIP,SourceIP][Interval,Min-Rx,Multiplier], ...
                        :
```

```
Max Path                : 8
Redistribute            : connected
================================================================================================
================================================================

efa tenant epg show
================================================================================================
================================================================
Name          : ten1epg1
Tenant        : ten1
Description   :
Type          : extension
Ports         :
POs           :
              : unstable            : ten1po1
Port Property : switchport mode     : trunk
              : native-vlan-tagging : false
NW Policy     : ctag-range          : 11-12
              : vrf                 : ten1vrf1 [unstable]
              : l3-vni              : 8191
Network Property [Flags : * - Native Vlan]
+------+--------+-------------+-------------+---------+-------------------------------
+-------------------------+-------------+-----------------------+---------------------
+-----------------+-----------+
| Ctag | L2-Vni | Anycast-IPv4 | Anycast-IPv6 | BD-name | Local IP (Device-IP->Local-IP)
|    Ctag-Description     | Mtu-IPv6-ND | ManagedConfig-IPv6-ND | OtherConfig-IPv6-ND
|    Dev-state    | App-state |
+------+--------+-------------+-------------+---------+-------------------------------
+-------------------------+-------------+-----------------------+---------------------
+-----------------+-----------+
| 11   | 11     | 10.0.11.1/24 |             |         |
| Tenant L3 Extended VLAN |             | False                 | False               |
not-provisioned | cfg-ready |
+------+--------+-------------+-------------+---------+-------------------------------
+-------------------------+-------------+-----------------------+---------------------
+-----------------+-----------+
| 12   | 12     | 10.0.12.1/24 |             |         |
| Tenant L3 Extended VLAN |             | False                 | False               |
not-provisioned | cfg-ready |
+------+--------+-------------+-------------+---------+-------------------------------
+-------------------------+-------------+-----------------------+---------------------
+-----------------+-----------+
For 'unstable' entities, run 'efa tenant po/vrf show' for details
================================================================================================
================================================================

efa tenant service bgp peer show
================================================================================================
================================================================================================
========
Name        : ten1bgppeer1
Tenant      : ten1
State       : bs-state-created
Description :
Static Peer:
+--------------+----------+------+---------+-----------+------------+-------------
+--------------+--------+----------------+---------------+------------------
+-----------------+-------------+
|  Device IP   |   VRF    | AFI  |  SAFI   | REMOTE IP | REMOTE ASN | BFD Enabled | BFD
Interval | BFD Rx | BFD Multiplier | Next Hop Self | Update Source IP |    Dev-state
|  App-state  |
+--------------+----------+------+---------+-----------+------------+-------------
+--------------+--------+----------------+---------------+------------------
+-----------------+-------------+
```

```
| 10.20.246.16 | ten1vrf1 | ipv4 | unicast | 10.1.0.0  | 65001      | false       |
0            | 0      | 0              | false         |                  |
provisioned     | cfg-in-sync |
+--------------+----------+------+---------+-----------+------------+-------------
+--------------+--------+----------------+---------------+------------------
+-----------------+-------------+
| 10.20.246.15 | ten1vrf1 | ipv4 | unicast | 10.0.0.0  | 65001      | false       |
0            | 0      | 0              | false         |                  | not-
provisioned | cfg-ready   |
+--------------+----------+------+---------+-----------+------------+-------------
+--------------+--------+----------------+---------------+------------------
+-----------------+-------------+
Dynamic Peer:
+-----------+-----+-----+------+-------------+-----------+-------------+-----------
+-----------+
| Device IP | VRF | AFI | SAFI | Listen Range | Peer Group | Listen Limit | Dev-state |
App-state |
+-----------+-----+-----+------+-------------+-----------+-------------+-----------
+-----------+
================================================================================================
================================================================================================
========
```

Scenario (Deletion of pre-provisioned config)	EFA 2.3.2	EFA 2.4.0
`efa tenant service bgp peer delete --name ten1bgppeer1 --tenant ten1`	`BgpService Deletion Failed: Device: 10.20.246.15 Bgp Peer Error: Device is administratively down`	`BgpService deleted successfully`
`efa tenant service bgp peer-group delete --name ten1bgppeergroup1 --tenant ten1`	`BgpService deletion Failed: Device: 10.20.246.15 Bgp Peer Error: Device is administratively down`	`BgpService deleted successfully`
`efa tenant epg delete --name ten1epg1 --tenant ten1`	`EndpointGroup deletion failed: Error: [10.20.246.15] device(s) are administratively down`	`EndpointGroup: ten1epg1 deleted successfully`
`efa tenant po delete --name ten1po1 --tenant ten1`	`Portchannel deletion failed: Error: [10.20.246.15] device(s) are administratively down`	`PortChannel: ten1po1 deleted successfully`

In-flight Transaction Recovery

EFA can recover in-flight (in-progress) transactions after a service restart or high-availability failover.

Overview

After a service restart or high-availability failover, EFA can recover in-flight transactions by rolling them back or rolling them forward. In-flight transactions are those that are outstanding in the execution log after a restart or a failover.

- When transactions are rolled back, the requested action is undone.
- When transactions are rolled forward, the requested action is completed.

By default, this feature enables the automatic recovery of Day-1 through Dan-N operations for tenant-related configurations. You can use the **`efa system feature update --inflight-transaction-auto-recovery disable`** command to disable the feature. When the feature is enabled, the recovery strategy is as follows.

Table 15: Recovery strategy

Operation type	Commands	Strategy
Create operations	• **`efa tenant create`** • **`efa tenant epg create`** • **`efa tenant po create`** • **`efa tenant service bgp peer create`** • **`efa tenant service bgp peer-group create`** • **`efa tenant vrf create`**	Roll back
Delete operations	• **`efa tenant delete`** • **`efa tenant epg delete`** • **`efa tenant po delete`** • **`efa tenant service bgp peer delete`** • **`efa tenant service bgp peer-group delete`** • **`efa tenant vrf delete`**	Roll forward

Table 15: Recovery strategy (continued)

Operation type	Commands	Strategy
Update with add operations, such as port-add, ctag-range-add, and vrf-add	• **efa tenant update** • **efa tenant epg update** • **efa tenant po update** • **efa tenant service bgp peer update** • **efa tenant service bgp peer-group update** • **efa tenant vrf update**	Roll back
Update with delete operations, such as port-delete, vrf-delete, and ctag-range-delete	• **efa tenant update** • **efa tenant epg update** • **efa tenant po update** • **efa tenant service bgp peer update** • **efa tenant service bgp peer-group update** • **efa tenant vrf update**	Roll forward

Consider the following expected behaviors for this feature:

- During operations that take a long time, such as drift and reconcile and firmware downloads, tenant operations and recovery operations are blocked.
- When multiple transactions are pending in the execution log after a restart or a failover, recovery occurs in the order in which the operations appear in the execution log.
- If a service restart or high availability failover occurs during transaction recovery, then the status of those recovery operations is changed to a normal status. For example, if a restart occurs during the rollback of an EPG, the status changes to delete-pending. There is no automatic recovery of interrupted recovery transactions. You must manually verify and address the status of such operations.

Important

Day-0 and administrative operations (those for the Inventory Service and Fabric Service) are not recovered automatically. If these operations are interrupted by a service restart or a failover, you must manually redo the operations.

Examples

This example enables automatic in-flight transaction recovery.

```
efa system feature update --inflight-transaction-auto-recovery enable
Feature Setting Updated Successful
--- Time Elapsed: 634.557118ms ---
```

This example disables automatic in-flight transaction recovery.

```
efa system feature update --inflight-transaction-auto-recovery disable
Feature Setting Updated Successful
--- Time Elapsed: 634.557125ms ---
```

Tenant “show” CLI Consistency

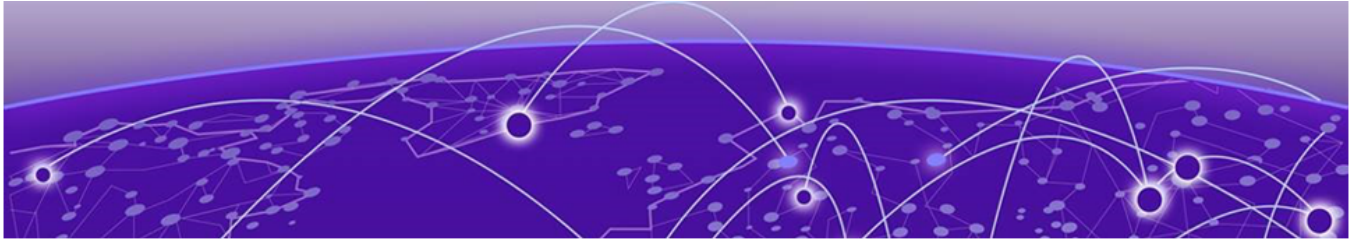
With the proposed changes, each entity (Tenant, PO, VRF, BGP Peer-Group, BGP Peer, EPG) will have the below possible "show" flavours:

Brief	Detail
• Displays the brief info of all the instances belonging to all the tenants in the format "efa tenant <entity-type> show" e.g. "efa tenant po show". • Displays the brief info of all the instances belonging to a given tenant in the format "efa tenant <entity-type> show --tenant <tenant-name>" e.g. "efa tenant po show --tenant ten1". • Displays the brief info of a particular instance in the format "efa tenant <entity-type> show --name <entity-instance-name> --tenant <tenant-name>". Example `efa tenant po show --name ten1po1 --tenant ten1`	• Displays the detailed info of all the instances belonging to all the tenants in the format "efa tenant <entity-type> show --detail" e.g. "efa tenant po show --detail" • Displays the detailed info of all the instances belonging to a given tenant in the format "efa tenant <entity-type> show --tenant <tenant-name> --detail" e.g. "efa tenant po show --tenant ten1 --detail" • Displays the detailed info of a particular instance in the format "efa tenant <entity-type> show --name <entity-instance-name> --tenant <tenant-name> --detail" Example `efa tenant po show --name ten1po1 --tenant ten1 --detail`

Scale and Performance

Test Case	EFA 2.3.2 GA	2.4 Build 19	2.4 Build 28 (CD2)	2.4 Build 35 (towards CD4)
Create epg with 5 ctags and 50 port-channels - Spaning across 6 mct pairs.	N/A	53.307872396s (for single network)	53-83s (for single network) 5m 28s (for five network)	32s-60s (for single network) 4m 09s (for five network)
Create an empty epg with single ctag, add port-channel via port-group update one port-channel at a time: create 5 network epgs, Add 50 port-channels, 250 epg update.	N/A	6m 50s (for single network 50 updates)	05m:15s (for single network 54 update)	04m:31s (for single network 54 update)
L3- single vrf epg with 5 ctag and 50 port-groups - different mct pairs - single command with vrf.	N/A	Not captured	134-173s (for single network) 12m:32s (for five network)	76s-137s (for single network) 10m:6s (for five network)

Test Case	EFA 2.3.2 GA	2.4 Build 19	2.4 Build 28 (CD2)	2.4 Build 35 (towards CD4)
L3- single vrf empty epg with ctag, add port-channel via port-group update one port-channel at a time: create 5 network epgs, Add 50 port-groups, 150 epg update - VRF update Remove port-groups.	N/A	Not captured	7m:14s (for single network 54 update)	6m:55s (for single network 54 update)
Create L3 EPG with ports across two nodes, one ctag.	16.419601664 s	13.464s	15.02s	N/A



EFA Device Management

Device Image Management

Using maintenance mode, you can download firmware on one or more devices in the IP fabric with minimal disruption to data path traffic. Both Clos and non-Clos fabrics are supported. The maintenance mode feature is supported only on SLX devices running SLXOS 20.1.1 and later.

EFA supports the following firmware download features.

- Firmware download with maintenance mode supporting the following:
  - Asynchronously launched operations
  - Sanity and pre-install script verification
  - Set convergence timeout, enable, and disable
  - Persisting the running configuration so that running configuration and maintenance mode configuration are preserved after reboot
  - Firmware download with the `no commit` option to enable restoration of firmware to a previous version
- Firmware host registration, with support for register, update, delete, and list operations
- Firmware download preparation, with support for add, remove, and list operations
- Firmware download with the `show` option, to display a table of devices in the fabric and their corresponding status

Limitations

- The device firmware must be SLX-OS 20.1.1 or later to support firmware download with maintenance mode for a hitless firmware upgrade.
- This feature assumes an existing host that contains SLX-OS firmware images ready to be downloaded.
- You can use this feature on a device where EFA TPVM is deployed, as long as you follow the instructions in Upgrade Device Firmware in a High Availability Deployment on page 153.
- If you downgrade software from version 20.1.2a to 20.1.1, you must manually remove certificates.

Supported devices

The SLX-OS firmware download with maintenance mode is supported on the following SLX devices running SLX-OS 20.1.1 and later.

- SLX 9540
- SLX 9640
- SLX 9150-48Y
- SLX 9150-48XT
- SLX 9250
- SLX 9740

Hitless Firmware Upgrade

A hitless firmware upgrade uses the maintenance mode feature of the SLX device to gracefully divert traffic away from the device to alternate paths. The device can be put into maintenance mode and a firmware upgrade can be performed. The device can safely be rebooted and the new firmware activated without traffic loss. When the device is taken out of maintenance mode, traffic is allowed on the newly upgraded device.

Upgrading the Super-Spine Firmware in Clos

1. The firmware on the first super-spine is downloaded.
2. Enabling maintenance mode on a super-spine involves the Border Gateway Protocol (BGP). The `graceful_shutdown` parameter is sent to all the super-spine's underlay neighbors (all connected spines). Each neighbor processes the `graceful_shutdown` and refreshes their routes to use the alternate path. Maintenance mode is enabled on the first super-spine and traffic is diverted to the second super-spine.
3. The `running-configuration` is saved on the first super-spine to preserve all current configurations including the maintenance mode enable configuration.
4. The device is rebooted for firmware activation without traffic loss.
5. Once the new firmware is activated, maintenance mode can be disabled. The `graceful_shutdown` parameter is removed from all the underlay neighbors and traffic to the first super-spine is allowed again.
6. The `running-config` is persisted again to ensure the maintenance mode disabled state is retained.

   The same process can be carried out on the second super-spine to upgrade the firmware without traffic loss.

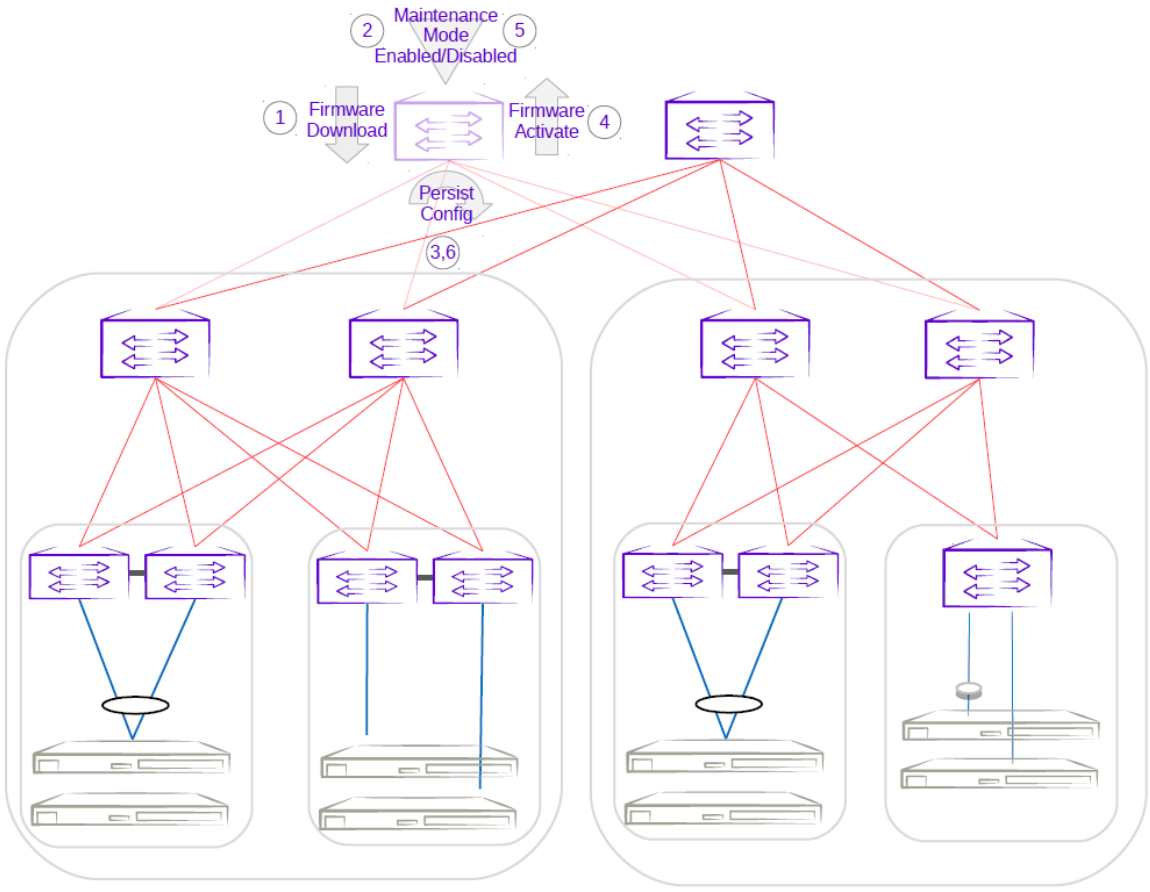


Figure 20: First super-spine firmware upgrade with maintenance mode

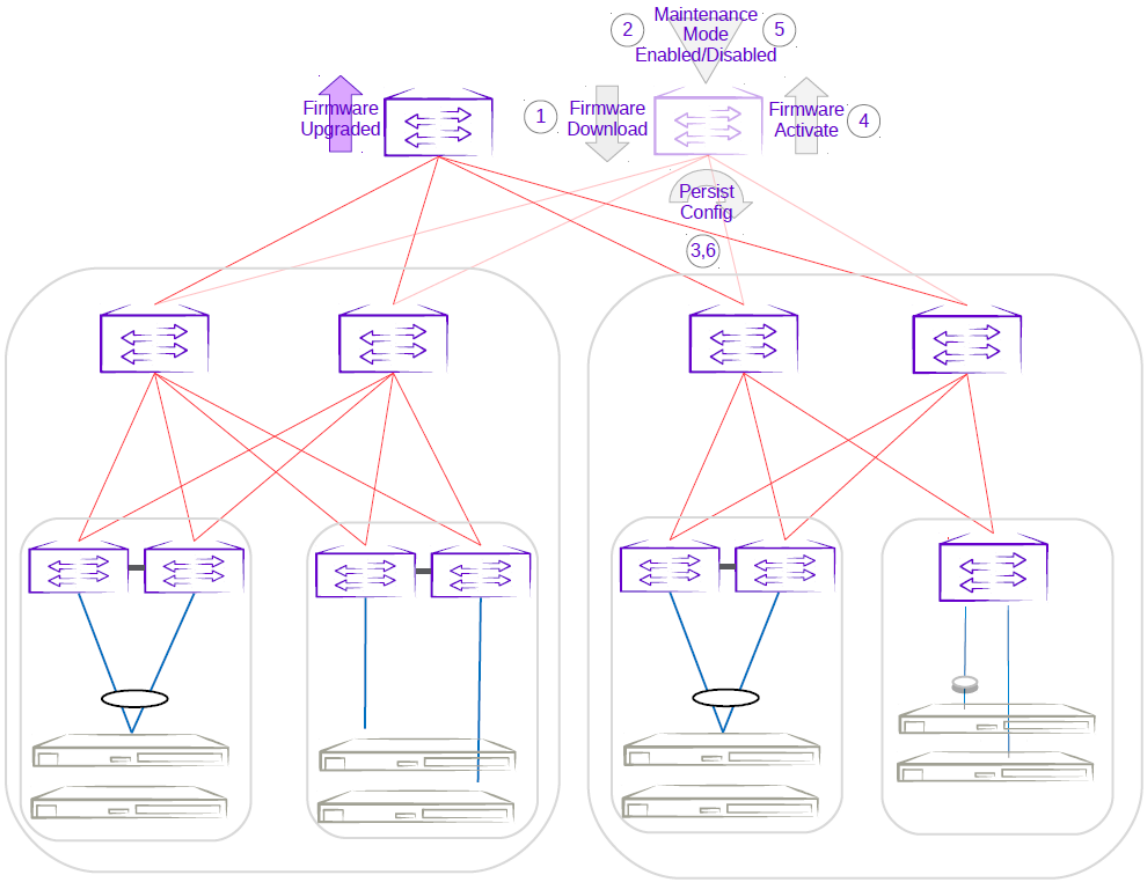


Figure 21: Second super-spine firmware upgrade with maintenance mode

Upgrading the Spine Firmware in Clos

1. The firmware on the first spine is downloaded.
2. Enabling maintenance mode on a spine also involves the Border Gateway Protocol (BGP). The `graceful_shutdown` parameter is sent to all the spine's underlay neighbors (all leafs in the pod and super-spines). The neighbors no longer send traffic to the first spine going into maintenance mode and redirect traffic to an alternate path.
3. The `running-configuration` is saved on the first spine to preserve all current configurations including the maintenance mode enable configuration.
4. The device is rebooted for firmware activation without traffic loss.
5. Once the new firmware is activated, maintenance mode is disabled to allow traffic again through the upgraded spine.
6. The `running-config` is saved again to ensure the maintenance mode config remains disabled.

   The same process can be carried out on the second spine to upgrade the firmware without traffic loss.

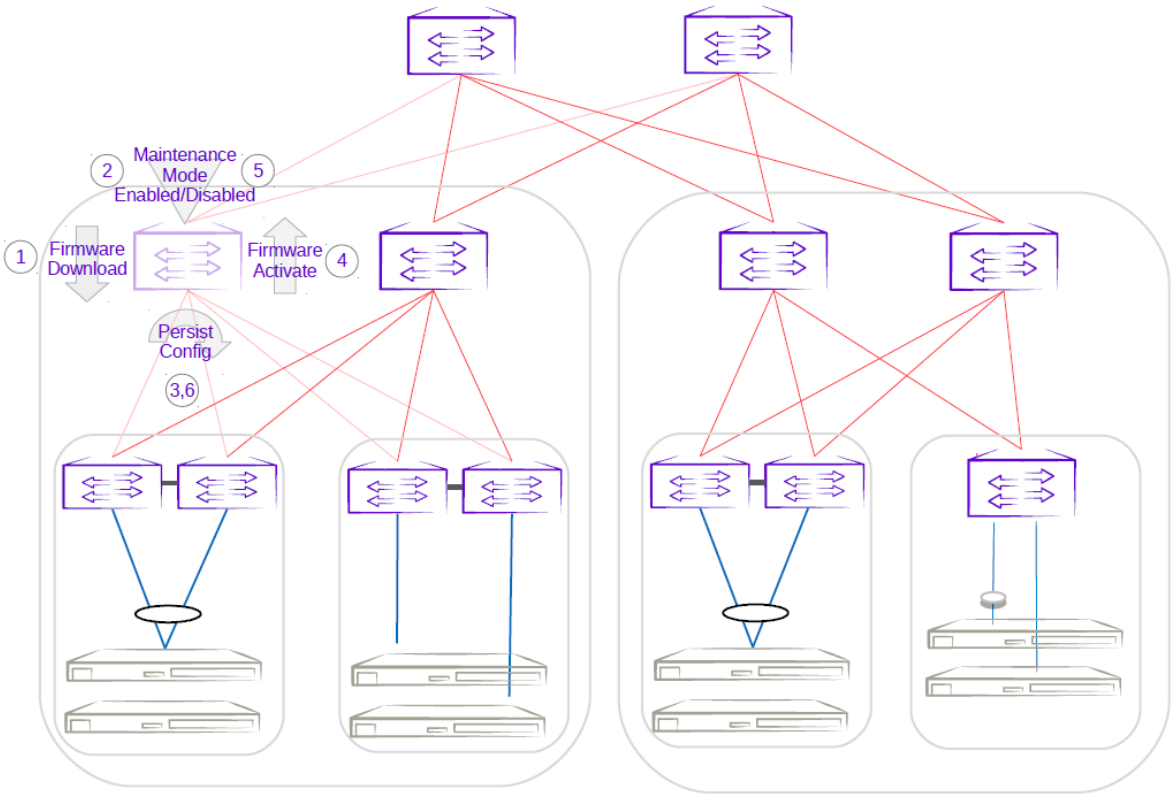


Figure 22: First spine firmware upgrade with maintenance mode

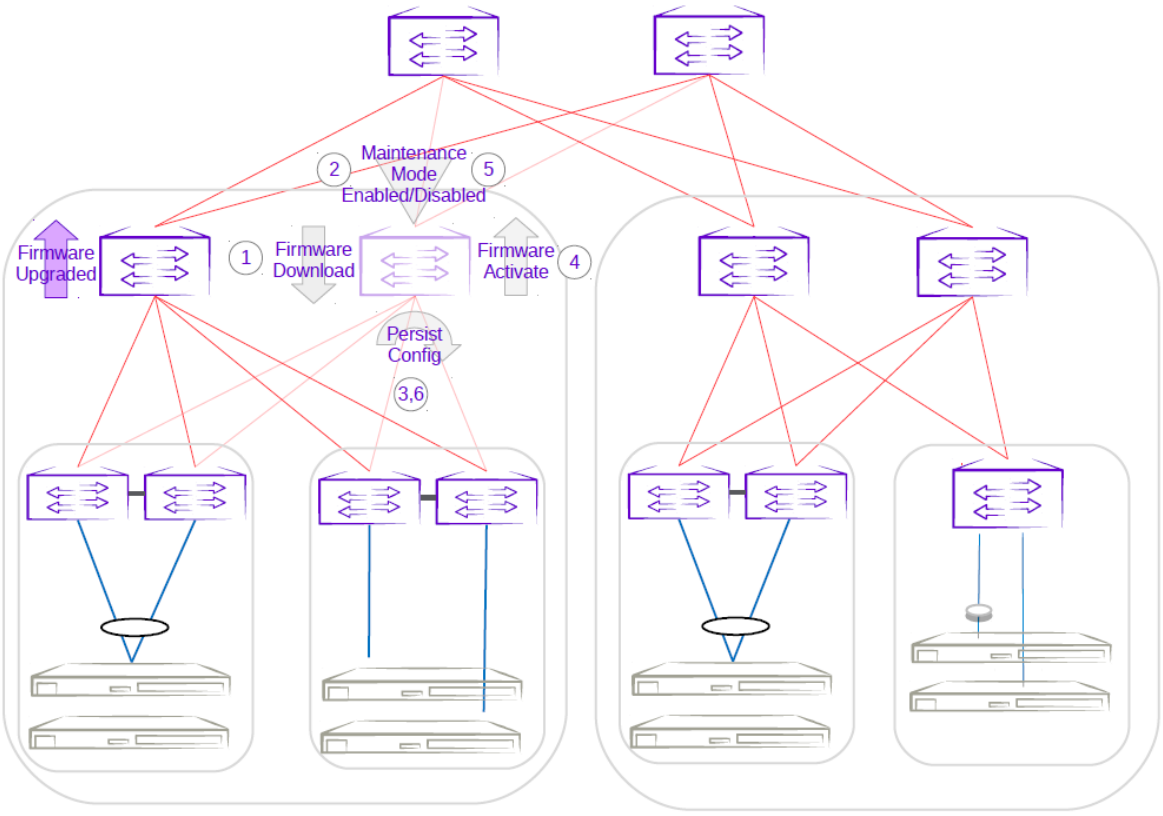


Figure 23: Second spine firmware upgrade with maintenance mode

Upgrading the Firmware of an MCT Leaf Pair with Dual-Homed Servers in Clos

1. The firmware on the MCT leaf is downloaded.
2. Enabling maintenance mode on an MCT leaf involves the Border Gateway Protocol (BGP) and MCT or NSM. The `graceful_shutdown` parameter is sent to all the leaf's underlay neighbors (all spines in the pod). The neighbors no longer send traffic to the MCT leaf going into maintenance mode and redirect traffic from spines to the peer MCT leaf. MCT instructs the peer leaf to become the designated forwarder, ICL is shut down, and CCE ports for clients are also shut down. Traffic from dual-homed servers is redirected to the peer leaf. With maintenance mode enabled, traffic is completely redirected to the peer leaf.
3. The `running-configuration` is saved on the first MCT leaf to preserve all current configurations including the maintenance mode enable configuration.
4. The device is rebooted for firmware activation without traffic loss.
5. After the firmware is upgraded, the maintenance mode is disabled to allow traffic again through the upgraded MCT leaf.
6. The `running-config` is saved again to ensure the maintenance mode config remains disabled.

   The same process can be carried out on the second MCT leaf to upgrade the firmware without traffic loss.

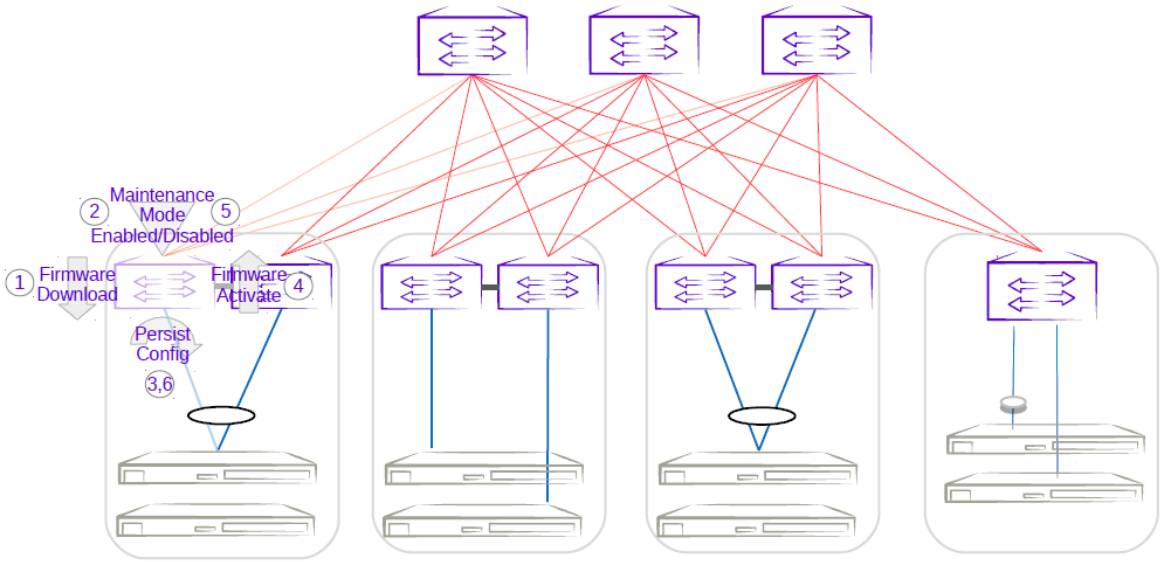


Figure 24: First MCT leaf firmware upgrade with maintenance mode

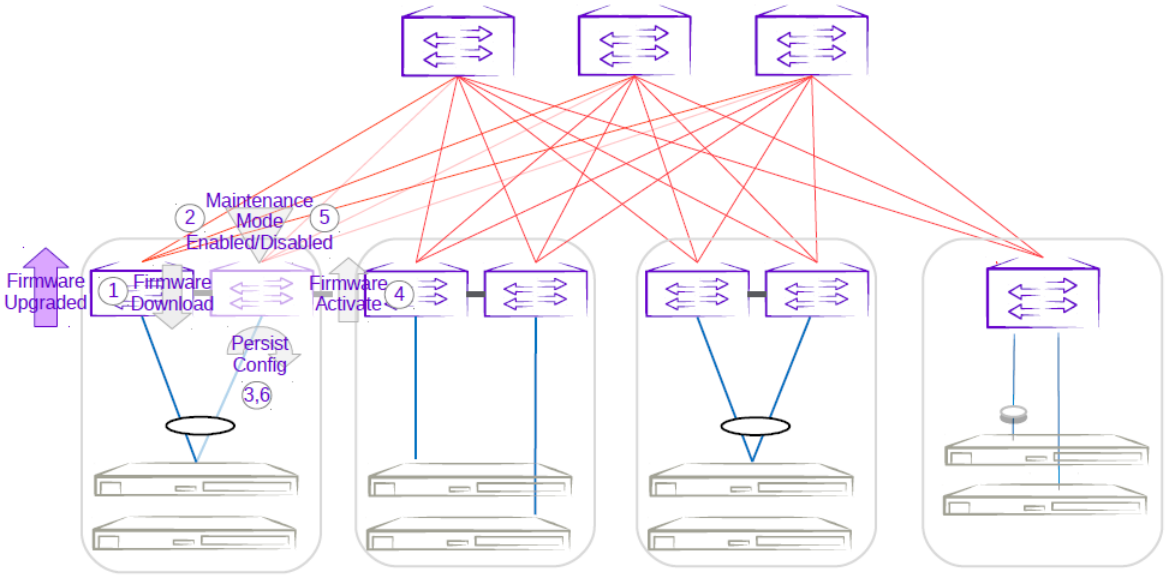


Figure 25: Second MCT leaf firmware upgrade with maintenance mode

Upgrading the Firmware of a Three-Rack Centralized MCT Pair in Non-Clos

1. The firmware on the MCT leaf is downloaded.
2. Enabling maintenance mode on one of the leafs in the centralized MCT leaf pair follows the same behavior as the MCT leaf pair in a Clos topology. The only difference is the iBGP Layer 3 backup link between MCT leaf pairs. Maintenance mode results in the traffic being redirected to the peer leaf in the centralized MCT leaf pairs.
3. The `running-configuration` is saved on the first MCT leaf to preserve all current configurations including the maintenance mode enable configuration.

4. The device is rebooted for firmware activation without traffic loss.
5. After the firmware is upgraded, the maintenance mode is disabled to allow traffic again through the upgraded MCT leaf.
6. The `running-config` is saved again to ensure the maintenance mode config remains disabled.

   The same process can be carried out on the second MCT leaf to upgrade the firmware without traffic loss.

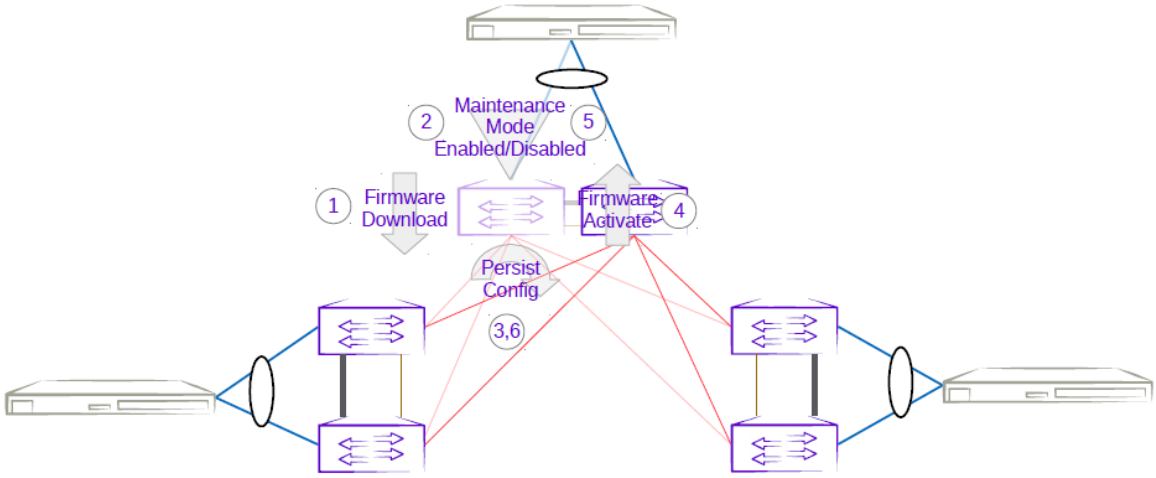


Figure 26: Three-rack centralized first MCT leaf firmware upgrade with maintenance mode

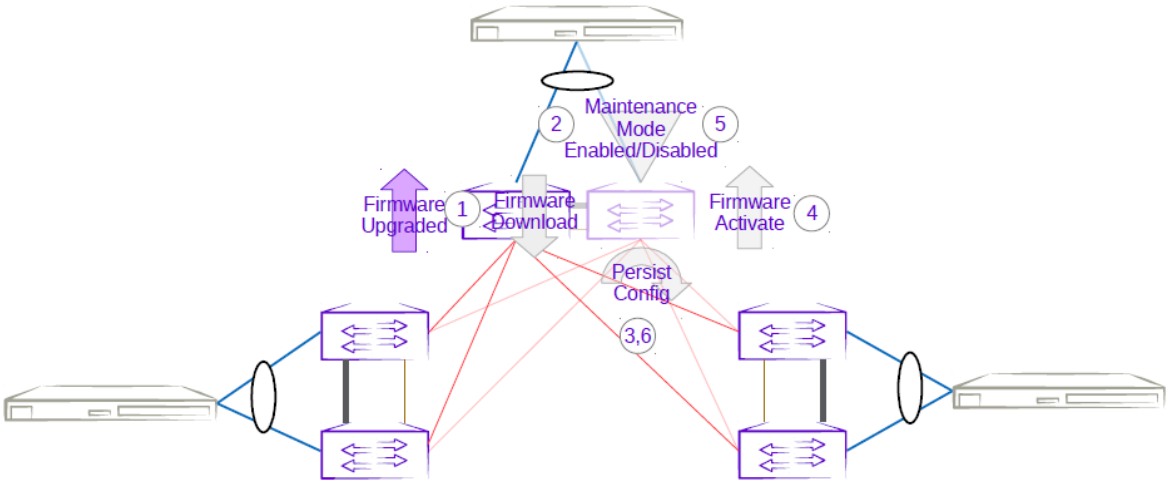


Figure 27: Three-rack centralized second MCT leaf firmware upgrade with maintenance mode

Upgrading the Firmware of a Three-Rack Ring MCT Pair in Non-Clos

1. The firmware on the MCT leaf is downloaded.

2. Enabling maintenance mode on one of the leafs in a three-rack ring MCT leaf pair follows the same behavior as the MCT leaf pair in a Clos topology. The only difference is the iBGP Layer 3 backup link between MCT leaf pairs. Maintenance mode results in the traffic being redirected to the peer MCT leaf.
3. The `running-configuration` is saved on the first MCT leaf to preserve all current configurations including the maintenance mode enable configuration.
4. The device is rebooted for firmware activation without traffic loss.
5. After the firmware is upgraded, the maintenance mode is disabled to allow traffic again through the upgraded MCT leaf.
6. The `running-config` is saved again to ensure the maintenance mode config remains disabled.

   The same process can be carried out on the second MCT leaf to upgrade the firmware without traffic loss.

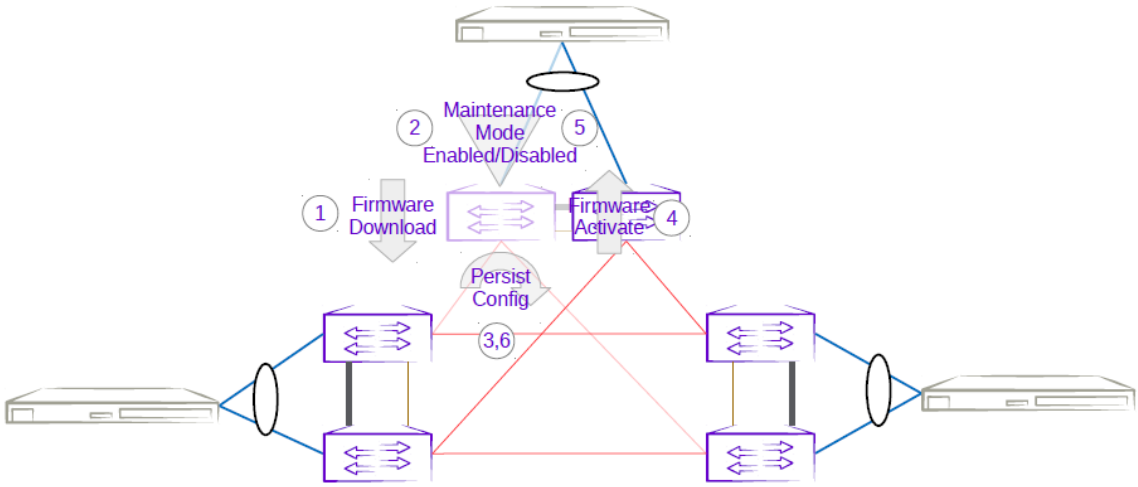


Figure 28: Three-rack ring first MCT leaf firmware upgrade with maintenance mode

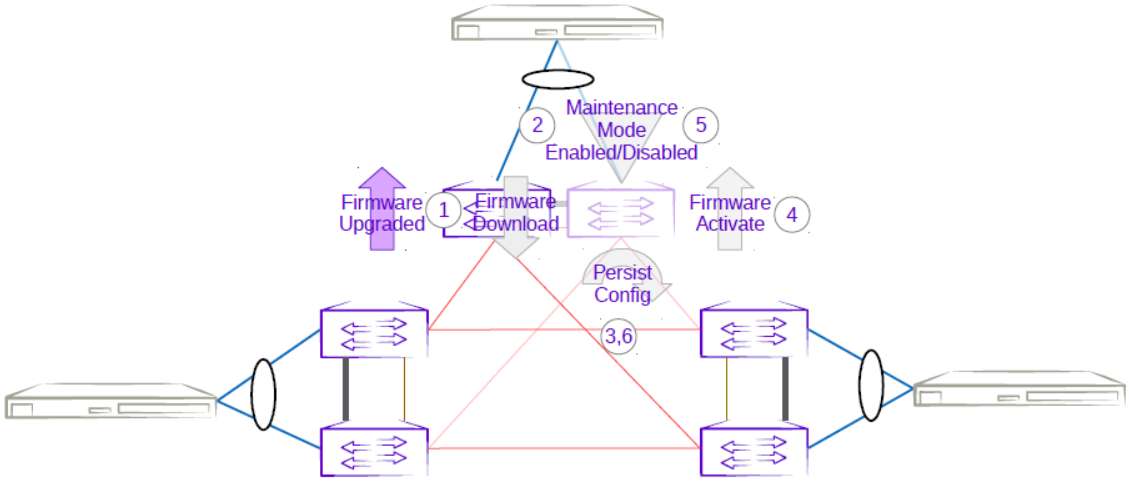


Figure 29: Three-rack ring second MCT leaf firmware upgrade with maintenance mode

Upgrade Device Firmware in a High Availability Deployment

Use this procedure to upgrade the firmware of SLX devices that host the high availability deployment on TPVMs.

This is the recommended method for upgrading the firmware of devices in a high-availability deployment. It describes how to upgrade the standby node, force a failover to change the active node to standby, and then upgrade the new standby node.

To upgrade firmware in all devices in the fabric, see Fabric-wide Firmware Download on page 154.

1. On the TPVM, determine which EFA node is the standby node.
   a. Run the EFA **`efactl status`** script (or the **`efa status`** command, as an alternative).

```
$ efactl status

NAME    STATUS   ROLES    AGE   VERSION        LABELS
tpvm    Ready    primary   21h   v1.18.6+k3s1   beta.kubernetes.io/arch=amd64,
beta.kubernetes.io/os=linux,keepalived=active,kubernetes.io/arch=amd64,
kubernetes.io/hostname=tpvm,kubernetes.io/os=linux,node-role.kubernetes.io/
primary=true

tpvm2   Ready    primary   21h   v1.18.6+k3s1   beta.kubernetes.io/arch=amd64,
beta.kubernetes.io/os=linux,keepalived=standby,kubernetes.io/arch=amd64,
kubernetes.io/hostname=tpvm2,kubernetes.io/os=linux,node-role.kubernetes.io/
primary=true
```

   The node that is labeled "keepalived=standby" is the standby node. The command also returns a list of EFA pods (not shown in the example). The node that runs the pods is the active node.
2. Prepare and run the firmware download on the device that hosts the standby node.
   a. Prepare the firmware download.

```
efa inventory device firmware-download prepare add --ip <device IP>
--firmware-host <IP of firmware download host> -
-firmware-directory <path to target firmware build>
```

   The command returns the following information in a table: IP address, host name, model, chassis name, ASN, role, current firmware, firware host, firmware directory, target firmware, and last update time.
   b. Download the firmware.

```
$ efa inventory device firmware-download execute --fabric <fabric name>
Firmware Download  [success]
```

   c. Monitor the progress of the firmware download.

```
$ efa inventory device firmware-download show --fabric <fabric name>

Don't run other commands on these devices until firmware download is in progress
--- Time Elapsed: 299.843244ms --
```

   d. Repeat step c until the firmware download is complete.

      Each time you repeat step c, the command returns a table that details the progress of the firmware download. The download is complete when the Update State column shows Completed and the Status column shows Firmware Committed.

3. Perform a high availability failover.
   a. On the device that hosts the active node, stop and start the TPVM to initiate a failover.

```
device# tpvm stop
stop succeeds

device# tpvm start
start succeeds
```

   After failover, the active node becomes the standby node.
   b. On the TPVM, validate that the failover is complete.

```
$ efactl status
```

   or

```
$ efa status
```

   The command returns a list of all pods and their metadata, including status, number of restarts, age, IP address, and node name.
4. Repeat steps 2a through 2d to upgrade the device firmware on the TPVM of the new standby node.

Note

If you do not follow the recommended steps and, instead, upgrade the firmware on the active node, then the EFA inventory becomes out of sync with the SLX device. The device remains in maintenance mode and the inventory indicates that the firmware download is in progress (even if it completed successfully). To resynchronize the inventory and the device, run the following commands to correct the device's firmware state in EFA and then to take the device out of maintenance mode.

```
$ efa inventory device update --ip <device IP>

$ efa inventory device setting update --maint-mode-enable No --ip <device IP>
```

Fabric-wide Firmware Download

Use this procedure to upgrade the firmware of SLX devices in a Clos fabric.

This is the recommended method for upgrading the firmware of devices in a Clos fabric. It describes how to upgrade the device of standby EFA node and MCT leaf pairs, force a failover to change the active node to standby, and then upgrade the SLX of new standby node and remaining MCT leaf pairs.

To upgrade firmware in a non-Clos configuration, see Upgrade Device Firmware in a High Availability Deployment on page 153.

1. On the TPVM, determine which EFA node is the standby node.
   a. Run the EFA **`efactl status`** script (or the **`efa status`** command, as an alternative).

```
$ efa status
+-----------+---------+--------+
| Node Name | Role    | Status |
+-----------+---------+--------+
| tpvm1     | active  | up     |
+-----------+---------+--------+
| tpvm2     | standby | up     |
+-----------+---------+--------+
```

2. Prepare and run the firmware download on the devices in the fabric, in batches. In batch-1, add the device that hosts the standby node and devices on right side of the fabric. The diagram that follows illustrates the right and left devices in the batches of a fabric.

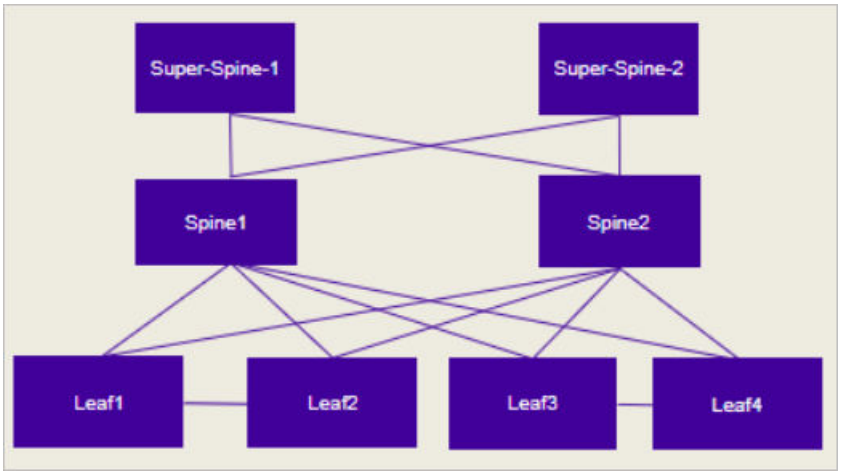


Figure 30: Batches for fabric-wide update

a. Prepare the firmware download.

```
$ efa inventory device firmware-download prepare add
--ip <device IPs separated by comma> --firmware-host <IP of firmware download host>
--firmware-directory <path to target firmware build>
```

The command returns the following information in a table: IP address, host name, model, chassis name, ASN, role, current firmware, firware host, firmware directory, target firmware, and last update time.

b. Download the firmware with or without the `–noAutoCommit` and `–noMaintMode` options, as desired.

```
$ efa inventory device firmware-download execute
--fabric <fabric name> --noAutoCommit –noMaintMode

Firmware Download Execute [success]

--noAutoCommit    Configure Auto commit in Firmware Download
--noMaintMode     Configure Maintenance Mode in Firmware Download
```

c. Monitor the progress of the firmware download.

```
$ efa inventory device firmware-download show
--fabric <fabric name>
```

d. Repeat step c until the firmware download is complete.

Each time you repeat step c, the command returns a table that details the progress of the firmware download. The download is complete when the Update State column shows **Completed** and the Status column shows **Firmware Not Committed** when `–noAutoCommit` is used and **Firmware Committed** without `–noAutoCommit`.

3. Perform a high availability failover.
   a. On the device that hosts the active node, stop and start the TPVM to initiate a failover.

```
device# tpvm stop
stop succeeds

device# tpvm start
start succeeds
```

   After failover, the active node becomes the standby node.
4. In batch-2, add the device that hosts the standby node (previously Active) and devices on the left side of the fabric. This should be executed from the current Active EFA. Repeat steps 2a through 2d to download firmware on devices in batch-2.

   The download is complete when the Update State column shows **Completed** on all devices. The Status column shows **Firmware Not Committed** when `–noAutoCommit` is used and **Firmware Committed** without `–noAutoCommit`.
5. Commit the firmware across all devices in the fabric.

```
$ efa inventory device firmware-download commit –fabric <fabric name>

OR

$ efa inventory device firmware-download commit –ip <IP address of all devices in
fabric>
```

   The download is complete when the Update State column shows **Completed** on all devices and the Status column shows **Firmware Committed**.

Roll Back Device Firmware

Firmware on the device can be rolled back when it is in "Firmware Not Committed" status.

This is the recommended method for rolling back firmware when it is not committed. Run firmware restore in batches on all devices in the fabric.

1. In batch-1, restore the firmware device that hosts EFA standby node and devices on the right side of the fabric.

```
$ efa inventory device firmware-download restore –ip <batch-1 device IPs separated by
comma >
```

   The download is complete when the Update State column shows **Completed** on all devices and the Status column shows **Firmware Rolled Back**.
2. Perform a high availability failover.
3. In batch-2, restore the firmware device that hosts EFA standby node (previously Active) and devices on the left side of the fabric.

```
$ efa inventory device firmware-download restore –ip <batch-2 device IPs separated by
comma >
```

   Note
   While fabric-wide restore is supported, traffic interruptions could still occur.

4. Commit the firmware across all devices in fabric.

```
$ efa inventory device firmware-download commit –fabric <fabric name>
```

Traffic Loss Scenarios

Single Leaf

Traffic loss is expected when you upgrade a single leaf that is not in an MCT pair. Because there are no alternate paths for the single leaf, maintenance mode is not enabled. Only the configuration is persisted, and a firmware upgrade is carried out. A traffic loss warning is flagged when you upgrade a single non-MCT leaf.

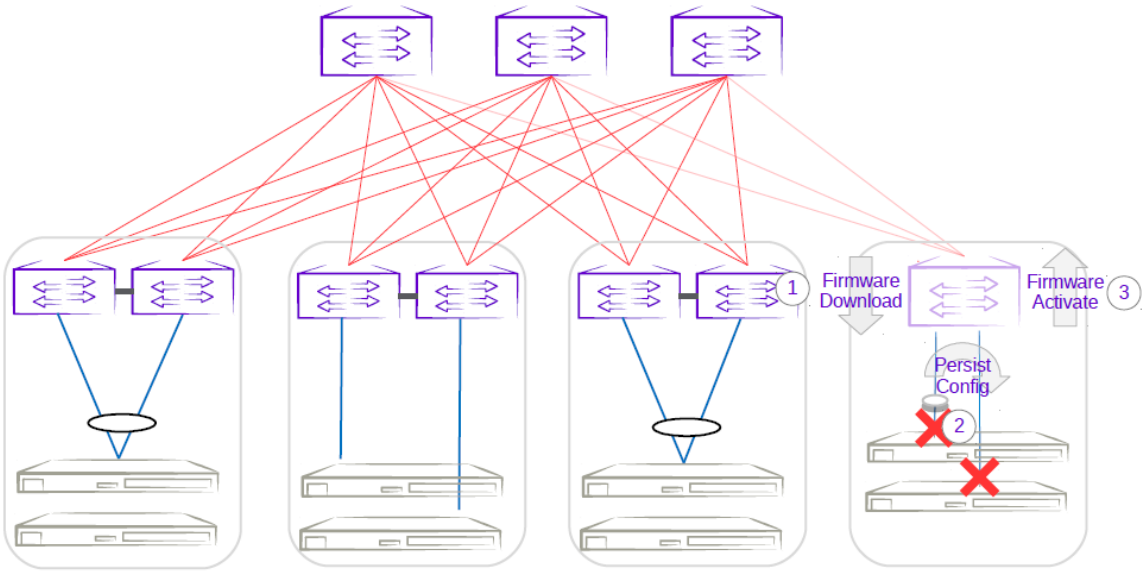


Figure 31: Single-leaf traffic loss

Single-Homed Server

Traffic loss is also expected for any singled-homed server. Detecting single-homed servers are not in the scope of this feature so a generic warning is provided at the start of a firmware download.

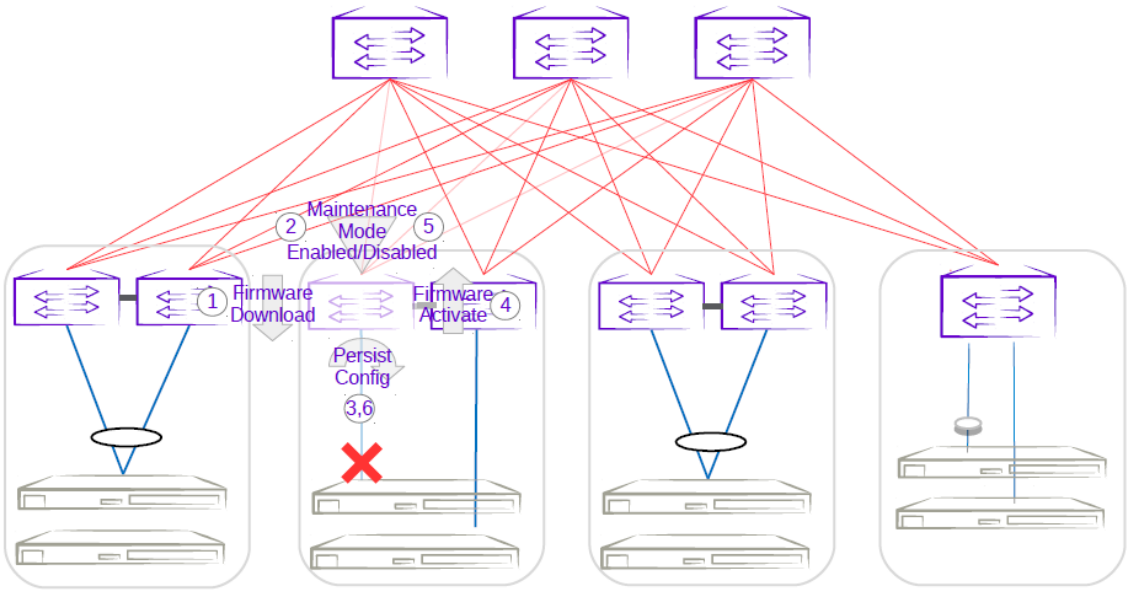


Figure 32: Single-homed server traffic loss

Non-Redundant Spine or Super-Spine

This is not a typical deployment, but traffic loss is expected in this scenario. Because no alternate paths exist for non-redundant switches, maintenance mode is not enabled for this case. A traffic loss warning is flagged when you upgrade non-redundant devices.

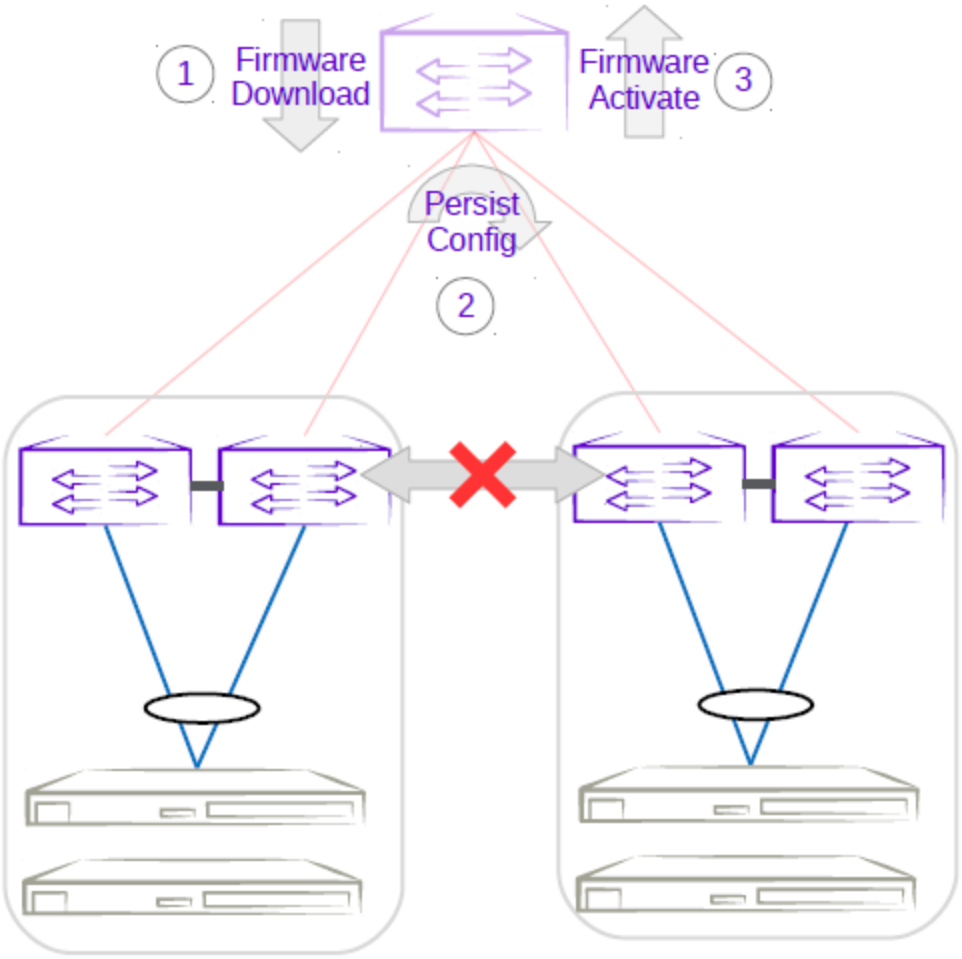


Figure 33: Non-redundant spine traffic loss

Switch Health Management

Switch Health Management (SHM) performs drift and reconciliation services, restoring fabric related configurations.

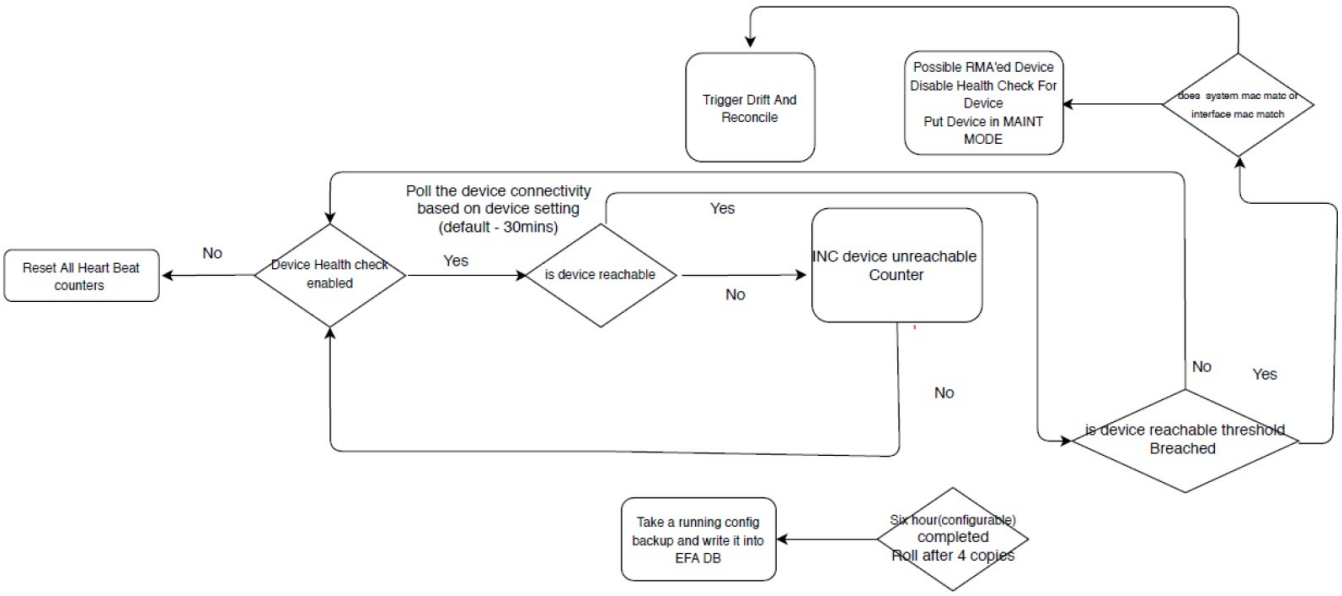


Figure 34: Switch Health Management work flow

Monitor Switch Health

The switches registered with EFA can be monitored for connectivity issues. If connectivity violates pre-defined thresholds, EFA starts drift and reconciliation.

1. Enable switch health check.

```
# efa inventory device setting update --ip 10.24.14.133 --health-check-enable yes
```

2. Configure health check interval.

```
# efa inventory device setting update --ip 10.24.14.133 --health-check-interval 30mins
```

3. Configure health check threshold.

```
# efa inventory device setting update --ip 10.24.14.133 --health-check-heartbeat-miss-threshold 2
```

4. View device health status.

```
# efa inventory device health status --ip 10.24.14.133
```

5. (Optional) Disable switch health check.

```
# efa inventory device setting update --ip 10.24.14.133 --health-check-enable no
```

Device Configuration Backup and Replay

The Device Configuration Backup and Replay feature enables backup of the device configuration based on inventory device settings or user-run commands and REST APIs.

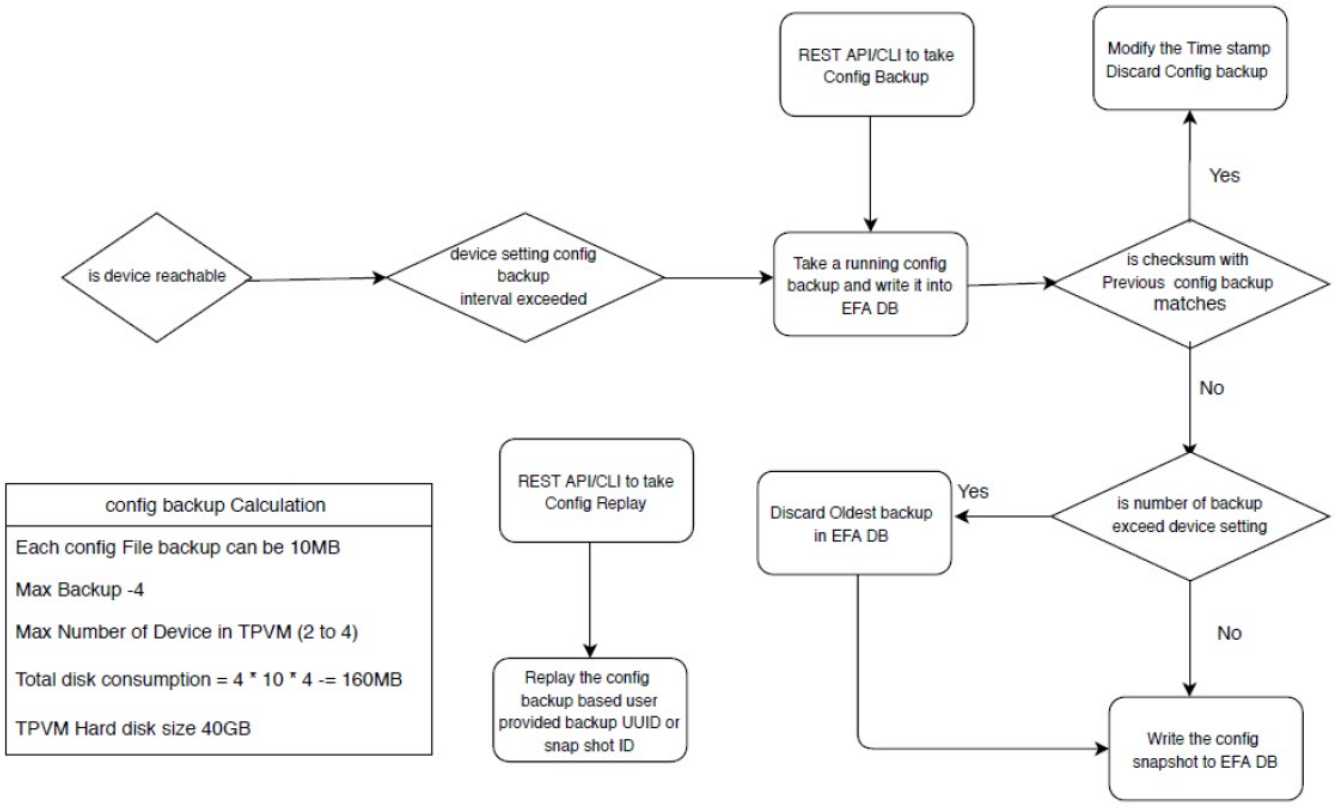


Figure 35: Workflow

Configure Backup and Replay

1. Enable periodic config-backup.

```
# efa inventory device setting update --ip 10.24.14.133 --config-backup-periodic-
enable yes
```

2. Configure device backup.

```
efa inventory device setting update --ip 10.24.14.133 --config-backup-interval 30m
[3m-1800m, default 1440m]
# efa inventory device setting update --ip 10.24.14.133 --number-of-config-backups 2
[2-20, default 4]
# efa inventory config-backup execute --ip 10.24.14.133
```

3. View config-backup history.

```
# efa inventory config-backup history --ip 10.24.14.133
# efa inventory config-backup detail --uuid 1111-1111-1111 --show-config
# efa inventory config-backup detail --uuid 1111-1111-1111 --show-config --file-dump
<filename>
```

4. (Optional) Delete config-backup.

```
# efa inventory config-backup delete --key 10.24.14.133
# efa inventory config-backup delete --key 1111-1111-111
```

5. Determine the backup restore method:
   - To restore backup using the `startup-config` file, proceed to the next step.
   - To restore backup using the `running-config` file, go to Step 7.

   Note
   The `startup-config` backup restore method requires device reboot to restore the configuration.

6. Configure device replay using the appropriate command.
   - Config-replay without rebooting the device:
     ```
     # efa inventory config-replay execute --ip 10.24.14.133 --uuid 1111-1111-111 --
     startup-config --no-reboot
     ```
   - Config-replay with device reboot:
     ```
     # efa inventory config-replay execute --ip 10.24.14.133 --uuid 1111-1111-111 --
     startup-config
     ```
7. Configure device replay using `running-config`.
   ```
   # efa inventory config-replay execute --ip 10.24.14.133 --uuid 1111-1111-111
   ```
8. View config-replay history.
   ```
   # efa inventory config-replay history --ip 10.24.14.133
   # efa inventory config-replay detail --uuid 1111-1111-1111
   ```
9. (Optional) Delete config-replay.
   ```
   # efa inventory config-replay delete --key 10.24.14.133
   # efa inventory config-replay delete --key 1111-1111-111
   ```

Return Material Authorization

With the Return Material Authorization (RMA) process, you can replace a faulty device with a new device that has the same configuration.

The high-level process is as follows. For specific steps and commands, see Replace a Faulty Device on page 164.

1. Verifying prerequisites.
   - Periodic configuration backup must be enabled on all devices that may need RMA. This prerequisite ensures that you have the latest configuration file to be used for recovery.
   - Maintenance mode must be enabled upon reboot on all devices.
2. Removing the faulty device and replacing it with the new device. Ports on this device must be administratively up and online.
3. Configuring the new device with the same management IP address and credentials as the old device.
4. Starting the RMA process from the command line or with the REST API.

   Note
   As a best practice, run the **`efa inventory rma execute`** command with the configuration backup ID so that the configuration is properly restored. If you run the command without the backup ID, you will need to manually update the configuration on the new device.

During the RMA process, the following actions occur:

- The device boots up in maintenance mode.
- EFA updates the device ID for the connection details in the database.
- Maintenance mode is initiated if the device is not already in maintenance mode.
- EFA replays the backed-up configuration specified by the `config-backup-id` parameter of the **`efa inventory rma execute`** command.
- EFA begins the drift reconcile process, which involves device discovery, device update, and fabric and tenant reconciliation. For more information, see Drift and Reconcile.

  Note
  If the RMA command fails during this stage, you can manually run the drift reconcile process from the CLI. If the RMA process fails for any other reason, restart the RMA process.

- When drift reconcile is complete, the device is taken out of maintenance mode.
- During the RMA process, EFA health checks are deactivated and RASlog does not trigger drift reconcile.

5. Installing the HTTPS or OAuth2 certificate on the new device.

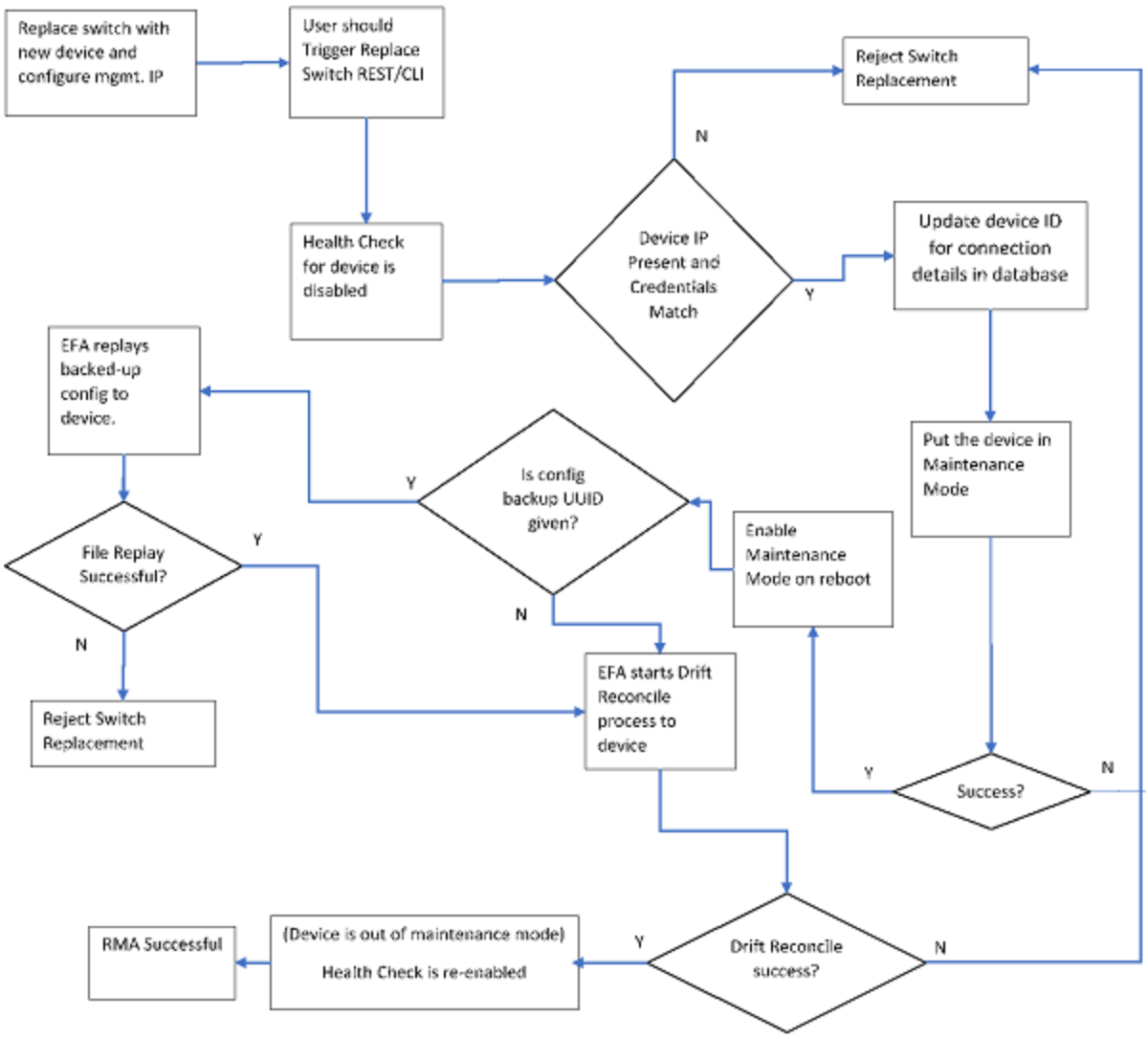


Figure 36: RMA work flow

Replace a Faulty Device

You can use the EFA command line to replace a faulty device with a new device and maintain the configuration of the old device.

Prerequisites:

- Ensure that periodic configuration backup is enabled on all devices that may need RMA. This prerequisite ensures that you have the latest configuration file to be used for recovery.
- Ensure that maintenance mode is enabled upon reboot on all devices.

This procedure describes how to replace a faulty device as part of the Return Material Authorization (RMA) process. For more information, see Return Material Authorization on page 162.

1. Obtain the configuration backup of the old device.

   For more information, see Configure Backup and Replay on page 161.

   ```
   # efa inventory config-backup execute --ip <ip-addr>
   ```

   This step generates a configuration backup ID that you use in step 5.
2. Replace the faulty device with the new device.
3. Ensure that the ports of the new device are administratively up and online.
4. Configure the new device with the same management IP address and credentials as the old device.
5. Start the RMA process.

   ```
   # efa inventory rma execute --ip <ip-addr> --config-backup-id <id>
   ```

   Note

   As a best practice, run the command with the `--config-backup-id <id>` option so that the configuration is properly restored. If you run the command without the backup ID, you will need to manually update the configuration on the new device.

6. View the RMA history and detail.

   ```
   # efa inventory rma history --ip <ip-addr>
   # efa inventory rma detail –uuid <uuid>
   ```

7. View the drift reconcile history and detail.

   ```
   # efa inventory drift-reconcile history --device-ip <ip-addr>
   # efa inventory drift-reconcile detail --uuid <uuid>
   ```

8. Install the HTTPS or OAuth2 certificate on the new device.

   ```
   # efa certificates device install --ips <device-ip-addr> --certType [https|token]
   ```

9. (Optional) Delete the RMA record.

   ```
   # efa inventory rma delete --ip <ip-addr>
   ```

10. (Optional) Manually start the drift reconcile process if the RMA process fails during the drift reconcile stage.

    ```
    # efa inventory drift-reconcile execute --ip <ip-addr> --reconcile
    ```

SLX Device Configuration

You can perform several configuration tasks on the SLX devices that EFA manages.

Compare a Device

You can view the configurations on the device that are out of sync with the configurations in the Asset service.

View a summary of the information in the Asset database.

```
efa inventory device compare --ip <IP address of the device>
```

Enable Maintenance Mode on SLX Devices

You can enable maintenance mode on the SLX devices that EFA manages.

By default, EFA performs Drift and Reconcile actions on the SLX devices that enter into maintenance mode after reboot, taking those devices out of maintenance mode after successfully reconciling the configuration on them. For more information about Drift and Reconcile, see Drift and Reconcile on page 39.

You can enable maintenance mode on SLX devices without triggering Drift and Reconcile. Take the following steps.

1. Disable syslog.

```
efa inventory device execute-cli --ip 10.18.120.187 --command "no logging syslog-
server 10.18.120.140 use-vrf mgmt-vrf" --config
```

2. Enable maintenance mode.

```
efa inventory device setting update --maint-mode-enable Yes --ip 10.18.120.187
```

The device remains in maintenance mode until you disable the mode.

If both `maint-mode-enable` and `maintenance-mode-enable-on-reboot` are set on the device, the Drift and Reconcile action is not triggered on device reboot.

3. Disable maintenance mode.
   a. Enable syslog.

```
efa inventory device execute-cli --ip 10.18.120.187 --command "logging syslog-
server 10.18.120.140 use-vrf mgmt-vrf" --config
```

   b. Run Drift and Reconcile.

```
efa inventory drift-reconcile execute --ip 10.18.120.187 -reconcile
```

The Drift and Reconcile process takes the device out of maintenance mode.

Configure Physical Port Speed

You can configure the speed for receiving and transmitting data on a physical port.

Tip

In SLX-OS, you can use the **`show interface ethernet`** command to see the current speed setting for the Ethernet interfaces on your device.

You can change the port speed for one or more IP addresses or for a specified fabric. For more configuration examples, see the *Extreme Fabric Automation Command Reference*.

Note

The **`efa inventory device interface set-speed`** command is an operational (or exec) command, not a configuration command. With operational commands, there is no configuration persistence, no drift identification, and no configuration reconciliation. You run operational commands as needed.

Run the **`efa inventory device interface set-speed`** command.

This example sets the port speed on multiple IP addresses.

```
efa inventory device interface set-speed --ip 10.25.225.167,10.24.48.131,
10.24.51.135 --if-name 0/20-22  --speed 25gbps
```

```
+---------------+-----+------+----------------+------------+---------+--------+
|   DeviceIP    | ID  | Name | Interface Type | Port Speed | Result  | Reason |
+---------------+-----+------+----------------+------------+---------+--------+
| 10.25.225.167 | 9   | 0/21 | ethernet       | 25gbps     | Success |        |
+               +-----+------+----------------+------------+---------+--------+
|               | 89  | 0/20 | ethernet       | 25gbps     | Success |        |
+               +-----+------+----------------+------------+---------+--------+
|               | 1   | 0/22 | ethernet       | 25gbps     | Success |        |
+---------------+-----+------+----------------+------------+---------+--------+
| 10.24.51.135  | 16  | 0/21 | ethernet       | 25gbps     | Success |        |
+               +-----+------+----------------+------------+---------+--------+
|               | 86  | 0/20 | ethernet       | 25gbps     | Success |        |
+               +-----+------+----------------+------------+---------+--------+
|               | 48  | 0/22 | ethernet       | 25gbps     | Success |        |
+---------------+-----+------+----------------+------------+---------+--------+
| 10.24.48.131  | 142 | 0/20 | ethernet       | 25gbps     | Success |        |
+               +-----+------+----------------+------------+---------+--------+
|               | 110 | 0/21 | ethernet       | 25gbps     | Success |        |
+               +-----+------+----------------+------------+---------+--------+
|               | 148 | 0/22 | ethernet       | 25gbps     | Success |        |
+---------------+-----+------+----------------+------------+---------+--------+
Interface Details
--- Time Elapsed: 53.8631425s ---
```

This example sets the port speed for the specified fabric.

```
efa inventory device interface set-speed --fabric nc_no_vni
--if-name 0/20-22 --speed 25gbps
+--------------+-----+------+----------------+------------+---------+--------+
|   DeviceIP   | ID  | Name | Interface Type | Port Speed | Result  | Reason |
+--------------+-----+------+----------------+------------+---------+--------+
| 10.24.51.135 | 86  | 0/20 | ethernet       | 25gbps     | Success |        |
+              +-----+------+----------------+------------+---------+--------+
|              | 48  | 0/22 | ethernet       | 25gbps     | Success |        |
+              +-----+------+----------------+------------+---------+--------+
|              | 16  | 0/21 | ethernet       | 25gbps     | Success |        |
+--------------+-----+------+----------------+------------+---------+--------+
| 10.24.48.131 | 142 | 0/20 | ethernet       | 25gbps     | Success |        |
+              +-----+------+----------------+------------+---------+--------+
|              | 110 | 0/21 | ethernet       | 25gbps     | Success |        |
+              +-----+------+----------------+------------+---------+--------+
|              | 148 | 0/22 | ethernet       | 25gbps     | Success |        |
+--------------+-----+------+----------------+------------+---------+--------+
Interface Details
--- Time Elapsed: 36.9974805s ---
```

Configure Breakout Ports

You can break a port into multiple interfaces, such as breaking one 40G port into four 10G ports. You can also revert the breakout.

In SLX-OS, you can use the **`show running-config hardware`** command to determine whether breakout mode is configured for a device.

You can break a port into the following modes: one 10g port, one 25g port, one 100g port, two 40g ports, two 50g ports, four 10g ports, and four 25g ports.

The breakout interfaces you create are identified by the name of the original interface followed by a suffix.

When you run revert a breakout, the breakout interfaces are deconfigured and deleted. The original Ethernet interface in the default configuration is created automatically.

You can configure breakout for one or more IP addresses or for a specified fabric. For more configuration examples, see the *Extreme Fabric Automation Command Reference*.

Note

The **efa inventory device interface set-breakout** command is an operational (or exec) command, not a configuration command. With operational commands, there is no configuration persistence, no drift identification, and no configuration reconciliation. You run operational commands as needed.

1. To break a port into multiple ports, run the **efa inventory device interface set-breakout** command.

   This example breaks three interfaces into four ports each.

```
efa inventory device interface set-breakout --ip 10.24.80.158
+--------------+----+-------+----------------+---------+--------+
|   DeviceIP   | ID | Name  | Interface Type | Result  | Reason |
+--------------+----+-------+----------------+---------+--------+
| 10.24.80.158 | 73 | 0/2:2 | ethernet       | Success |        |
+              +----+-------+----------------+---------+--------+
|              | 72 | 0/1:4 | ethernet       | Success |        |
+              +----+-------+----------------+---------+--------+
|              | 74 | 0/3:2 | ethernet       | Success |        |
+              +----+-------+----------------+---------+--------+
|              | 78 | 0/3:3 | ethernet       | Success |        |
+              +----+-------+----------------+---------+--------+
|              | 75 | 0/3:4 | ethernet       | Success |        |
+              +----+-------+----------------+---------+--------+
|              | 70 | 0/1:1 | ethernet       | Success |        |
+              +----+-------+----------------+---------+--------+
|              | 71 | 0/1:3 | ethernet       | Success |        |
+              +----+-------+----------------+---------+--------+
|              | 80 | 0/2:1 | ethernet       | Success |        |
+              +----+-------+----------------+---------+--------+
|              | 79 | 0/1:2 | ethernet       | Success |        |
+              +----+-------+----------------+---------+--------+
|              | 76 | 0/2:3 | ethernet       | Success |        |
+              +----+-------+----------------+---------+--------+
|              | 69 | 0/3:1 | ethernet       | Success |        |
+              +----+-------+----------------+---------+--------+
|              | 77 | 0/2:4 | ethernet       | Success |        |
+--------------+----+-------+----------------+---------+--------+
Interface Details
--- Time Elapsed: 48.3801684s ---
```

   This example configures break out for the specified fabric.

```
efa inventory device interface set-breakout --if-name 0/19-20
--mode 4x25g --fabric fabric1
+--------------+-----+--------+----------------+---------+--------+
|   DeviceIP   | ID  |  Name  | Interface Type | Result  | Reason |
+--------------+-----+--------+----------------+---------+--------+
| 10.24.80.158 | 188 | 0/20:2 | ethernet       | Success |        |
+              +-----+--------+----------------+---------+--------+
|              | 184 | 0/20:3 | ethernet       | Success |        |
+              +-----+--------+----------------+---------+--------+
|              | 190 | 0/20:4 | ethernet       | Success |        |
+              +-----+--------+----------------+---------+--------+
|              | 191 | 0/19:1 | ethernet       | Success |        |
```

```
+               +-----+--------+----------------+---------+--------+
|               | 187 | 0/19:2 | ethernet       | Success |        |
+               +-----+--------+----------------+---------+--------+
|               | 189 | 0/19:3 | ethernet       | Success |        |
+               +-----+--------+----------------+---------+--------+
|               | 185 | 0/19:4 | ethernet       | Success |        |
+               +-----+--------+----------------+---------+--------+
|               | 186 | 0/20:1 | ethernet       | Success |        |
+---------------+-----+--------+----------------+---------+--------+
| 10.24.80.159 | 196 | 0/19:2 | ethernet       | Success |        |
+               +-----+--------+----------------+---------+--------+
|               | 194 | 0/19:3 | ethernet       | Success |        |
+               +-----+--------+----------------+---------+--------+
|               | 197 | 0/19:4 | ethernet       | Success |        |
+               +-----+--------+----------------+---------+--------+
|               | 199 | 0/20:1 | ethernet       | Success |        |
+               +-----+--------+----------------+---------+--------+
|               | 193 | 0/20:2 | ethernet       | Success |        |
+               +-----+--------+----------------+---------+--------+
|               | 195 | 0/20:3 | ethernet       | Success |        |
+               +-----+--------+----------------+---------+--------+
|               | 192 | 0/20:4 | ethernet       | Success |        |
+               +-----+--------+----------------+---------+--------+
|               | 198 | 0/19:1 | ethernet       | Success |        |
+---------------+-----+--------+----------------+---------+--------+
Interface Details
--- Time Elapsed: 1m6.2210288s ---
```

2. To revert the breakout of multiple ports to the original configuration, run the **`efa inventory device interface unset-breakout`** command.

   This example removes breakout mode on multiple devices.

```
efa inventory device interface unset-breakout
--ip 10.24.80.158,10.24.80.159 --if-name 0/9-12
+--------------+--------------+----------------+----------------+---------+
|   DeviceIP   | Interface ID | Interface Name | Interface Type | Result  |
+--------------+--------------+----------------+----------------+---------+
| 10.24.80.158 | 248          | 0/10           | ethernet       | Success |
+              +--------------+----------------+----------------+---------+
|              | 250          | 0/11           | ethernet       | Success |
+              +--------------+----------------+----------------+---------+
|              | 249          | 0/12           | ethernet       | Success |
+              +--------------+----------------+----------------+---------+
|              | 247          | 0/9            | ethernet       | Success |
+--------------+--------------+----------------+----------------+---------+
| 10.24.80.159 | 252          | 0/10           | ethernet       | Success |
+              +--------------+----------------+----------------+---------+
|              | 254          | 0/11           | ethernet       | Success |
+              +--------------+----------------+----------------+---------+
|              | 253          | 0/12           | ethernet       | Success |
+              +--------------+----------------+----------------+---------+
|              | 251          | 0/9            | ethernet       | Success |
+--------------+--------------+----------------+----------------+---------+
Interface Details
--- Time Elapsed: 1m52.8562333s ---
```

   This example removes breakout mode for the specified Fabric name.

```
efa inventory device interface unset-breakout --if-name 0/19-20
--fabric fabric1
+--------------+--------------+----------------+----------------+---------+
|   DeviceIP   | Interface ID | Interface Name | Interface Type | Result  |
+--------------+--------------+----------------+----------------+---------+
| 10.24.80.159 | 279          | 0/19           | ethernet       | Success |
```

```
+              +--------------+----------------+----------------+---------+
|              | 280          | 0/20           | ethernet       | Success |
+--------------+--------------+----------------+----------------+---------+
| 10.24.80.158 | 281          | 0/19           | ethernet       | Success |
+              +--------------+----------------+----------------+---------+
|              | 282          | 0/20           | ethernet       | Success |
+--------------+--------------+----------------+----------------+---------+
Interface Details
--- Time Elapsed: 1m19.226463s ---
```

Configure MTU at the Interface or System Level

You can configure the MTU (maximum transmission unit) at the system level or at the physical port level for Layer 2, IPv4, and IPv6.

Tip

In SLX-OS, you can use the **show interface ethernet** command to see the current MTU configuration for an interface.

You can change the MTU for one or more IP addresses or for a specified fabric. For more configuration examples, see the *Extreme Fabric Automation Command Reference*.

Note

The **efa inventory device interface set-mtu** command is an operational (or exec) command, not a configuration command. With operational commands, there is no configuration persistence, no drift identification, and no configuration reconciliation. You run operational commands as needed.

1. At the interface level, run the **efa inventory device interface set-mtu** command.

   This example configures the MTU on multiple IP addresses.

```
efa inventory device interface set-mtu --ip 10.25.225.167,10.24.48.131,
10.24.51.135 --if-name 0/20-22  --mtu 3600 --ip-mtu 3600 --ipv6-mtu 3600
+---------------+-----+------+----------------+------+--------+----------+---------
|   DeviceIP    | ID  | Name | Interface Type | MTU  | IP MTU | IPv6 MTU | Result
+---------------+-----+------+----------------+------+--------+----------+---------
| 10.25.225.167 | 9   | 0/21 | ethernet       | 3600 | 3600   | 3600     | Success
+               +-----+------+----------------+------+--------+----------+---------
|               | 89  | 0/20 | ethernet       | 3600 | 3600   | 3600     | Success
+               +-----+------+----------------+------+--------+----------+---------
|               | 1   | 0/22 | ethernet       | 3600 | 3600   | 3600     | Success
+---------------+-----+------+----------------+------+--------+----------+---------
| 10.24.48.131  | 142 | 0/20 | ethernet       | 3600 | 3600   | 3600     | Success
+               +-----+------+----------------+------+--------+----------+---------
|               | 148 | 0/22 | ethernet       | 3600 | 3600   | 3600     | Success
+               +-----+------+----------------+------+--------+----------+---------
|               | 110 | 0/21 | ethernet       | 3600 | 3600   | 3600     | Success
+---------------+-----+------+----------------+------+--------+----------+---------
| 10.24.51.135  | 16  | 0/21 | ethernet       | 3600 | 3600   | 3600     | Success
+               +-----+------+----------------+------+--------+----------+---------
|               | 48  | 0/22 | ethernet       | 3600 | 3600   | 3600     | Success
+               +-----+------+----------------+------+--------+----------+---------
|               | 86  | 0/20 | ethernet       | 3600 | 3600   | 3600     | Success
+---------------+-----+------+----------------+------+--------+----------+---------
Interface MTU Details
--- Time Elapsed: 59.5462548s ---
```

This example configures the MTU for the specified Fabric name.

```
efa inventory device interface set-mtu --fabric nc_no_vni --if-name 0/20-22,0/55-58
--mtu 3200 --ip-mtu 3200 --ipv6-mtu 3200
+--------------+-----+---------+----------------+------+--------+----------+---------
+--------------------------
|   DeviceIP   | ID  |  Name   | Interface Type | MTU  | IP MTU | IPv6 MTU | Result
|       Reason
+--------------+-----+---------+----------------+------+--------+----------+---------
+--------------------------
| 10.24.51.135 | 16  | 0/21    | ethernet       | 3200 | 3200   | 3200     | Success
|
+              +-----+---------+----------------+------+--------+----------+---------
+--------------------------
|              | 48  | 0/22    | ethernet       | 3200 | 3200   | 3200     | Success
|
+              +-----+---------+----------------+------+--------+----------+---------
+--------------------------
|              | 86  | 0/20    | ethernet       | 3200 | 3200   | 3200     | Success
|
+              +-----+---------+----------------+------+--------+----------+---------
+--------------------------
|              | 0   | 0/55-58 | ethernet       | 0    | 0      | 0        | Failed  |
Interfaces [0/55-58]
|              |     |         |                |      |        |          |         |
do not exist for
                                                      device IP 10.24.51.135

Specify a valid interface
+--------------+-----+---------+----------------+------+--------+----------+---------
+--------------------------
| 10.24.48.131 | 0   | 0/55-58 | ethernet       | 0    | 0      | 0        | Failed  |
Interfaces [0/55-58]
|              |     |         |                |      |        |          |         |
does not exist for
                                                      device IP 10.24.51.131

Specify a valid interface
+              +-----+---------+----------------+------+--------+----------+---------
+--------------------------
|              | 110 | 0/21    | ethernet       | 3200 | 3200   | 3200     | Success
|
+              +-----+---------+----------------+------+--------+----------+---------
+--------------------------
|              | 148 | 0/22    | ethernet       | 3200 | 3200   | 3200     | Success
|
+              +-----+---------+----------------+------+--------+----------+---------
+--------------------------
|              | 142 | 0/20    | ethernet       | 3200 | 3200   | 3200     | Success
|
+--------------+-----+---------+----------------+------+--------+----------+---------
+--------------------------
Interface MTU Details
--- Time Elapsed: 37.3021602s ---
```

2. At the system level, run the **`efa inventory device system set-mtu`** command.

   This example configures the MTU on multiple IP addresses.

```
efa inventory device system set-mtu
--ip 10.25.225.167,10.24.48.131,10.24.51.135 --mtu 3600 --ip-mtu 3600
--ipv6-mtu 3600
WARNING: Setting MTU at device level will set MTU for all the interfaces
where its not explicitly set. This could cause some issues where traffic
is running. Do you want to proceed [y/n]?
y
```

```
Global MTU Updated Successfully
+----+---------------+------+--------+----------+---------+--------+
| ID |  IP Address   | MTU  | IP MTU | IPv6 MTU | Result  | Reason |
+----+---------------+------+--------+----------+---------+--------+
| 2  | 10.24.51.135  | 3600 | 3600   | 3600     | Success |        |
+----+---------------+------+--------+----------+---------+--------+
| 1  | 10.25.225.167 | 3600 | 3600   | 3600     | Success |        |
+----+---------------+------+--------+----------+---------+--------+
| 3  | 10.24.48.131  | 3600 | 3600   | 3600     | Success |        |
+----+---------------+------+--------+----------+---------+--------+
Global MTU Details
--- Time Elapsed: 50.7948311s ---
```

This example configures the MTU for the specified Fabric name.

```
efa inventory device system set-mtu --fabric nc_no_vni --mtu 3100
--ip-mtu 3100 --ipv6-mtu 3100
WARNING: Setting MTU at device level will set MTU for all the interfaces
where its not explicitly set. This could cause some issues where traffic
is running. Do you want to proceed [y/n]?
y
Global MTU Updated Successfully
+----+--------------+------+--------+----------+---------+--------+
| ID |  IP Address  | MTU  | IP MTU | IPv6 MTU | Result  | Reason |
+----+--------------+------+--------+----------+---------+--------+
| 2  | 10.24.51.135 | 3100 | 3100   | 3100     | Success |        |
+----+--------------+------+--------+----------+---------+--------+
| 3  | 10.24.48.131 | 3100 | 3100   | 3100     | Success |        |
+----+--------------+------+--------+----------+---------+--------+
Global MTU Details
--- Time Elapsed: 38.1473365s ---
```

Change the Admin Status of an Interface

You can bring an interface administratively up or down.

Tip

In SLX-OS, you can use the **show interface ethernet** command to see the status of the Ethernet interfaces on your device.

You can change the Admin Status for one or more IP addresses or for a specified fabric. For more configuration examples, see the *Extreme Fabric Automation Command Reference*.

Note

The **efa inventory device interface set-admin-state** command is an operational (or exec) command, not a configuration command. With operational commands, there is no configuration persistence, no drift identification, and no configuration reconciliation. You run operational commands as needed.

Run the **efa inventory device interface set-admin-state** command.

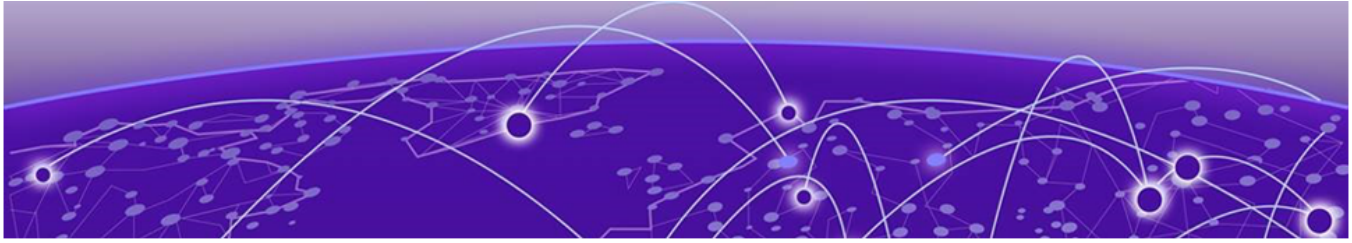
This example changes the Admin State on multiple IP addresses.

```
efa inventory device interface set-admin-state
--ip 10.25.225.167,10.24.48.131,10.24.51.135 --if-name 0/20-22 --state up

+---------------+-----+------+----------------+--------------+---------+--------+
|   DeviceIP    | ID  | Name | Interface Type | Admin Status | Result  | Reason |
+---------------+-----+------+----------------+--------------+---------+--------+
```

```
| 10.24.48.131  | 110 | 0/21 | ethernet       | up           | Success |         |
+               +-----+------+----------------+--------------+---------+---------+
|               | 142 | 0/20 | ethernet       | up           | Success |         |
+               +-----+------+----------------+--------------+---------+---------+
|               | 148 | 0/22 | ethernet       | up           | Success |         |
+---------------+-----+------+----------------+--------------+---------+---------+
| 10.24.51.135  | 16  | 0/21 | ethernet       | up           | Success |         |
+               +-----+------+----------------+--------------+---------+---------+
|               | 48  | 0/22 | ethernet       | up           | Success |         |
+               +-----+------+----------------+--------------+---------+---------+
|               | 86  | 0/20 | ethernet       | up           | Success |         |
+---------------+-----+------+----------------+--------------+---------+---------+
| 10.25.225.167 | 9   | 0/21 | ethernet       | up           | Success |         |
+               +-----+------+----------------+--------------+---------+---------+
|               | 89  | 0/20 | ethernet       | up           | Success |         |
+               +-----+------+----------------+--------------+---------+---------+
|               | 1   | 0/22 | ethernet       | up           | Success |         |
+---------------+-----+------+----------------+--------------+---------+---------+
Interface Details
--- Time Elapsed: 56.4964544s ---
```

This example changes the Admin Status for the specified fabric.

```
efa inventory device interface set-admin-state --fabric nc_no_vni
--if-name 0/20-22 --state up
+--------------+-----+------+----------------+--------------+---------+---------+
|   DeviceIP   | ID  | Name | Interface Type | Admin Status | Result  | Reason |
+--------------+-----+------+----------------+--------------+---------+---------+
| 10.24.51.135 | 48  | 0/22 | ethernet       | up           | Success |         |
+              +-----+------+----------------+--------------+---------+---------+
|              | 16  | 0/21 | ethernet       | up           | Success |         |
+              +-----+------+----------------+--------------+---------+---------+
|              | 86  | 0/20 | ethernet       | up           | Success |         |
+--------------+-----+------+----------------+--------------+---------+---------+
| 10.24.48.131 | 148 | 0/22 | ethernet       | up           | Success |         |
+              +-----+------+----------------+--------------+---------+---------+
|              | 142 | 0/20 | ethernet       | up           | Success |         |
+              +-----+------+----------------+--------------+---------+---------+
|              | 110 | 0/21 | ethernet       | up           | Success |         |
+--------------+-----+------+----------------+--------------+---------+---------+
Interface Details
--- Time Elapsed: 37.9236481s ---
```



EFA Event Management

RASlog Service

The RASlog Service is aware of all devices that are registered with the services in EFA and processes events only from those devices. Messages from other devices are dropped.

The RASlog Service performs the following functions:

- Acts as a syslog server to process syslog messages from devices
- Acts as an SNMP trap receiver to process traps from devices

With the RASlog Service, EFA receives events from network devices and the Inventory service learns of relevant changes. The Inventory Service can fetch the current state of network topology and update Fabric and Tenant services.

RASlog Operations

EFA is registered as a syslog recipient on the devices as part of the device registration. If there are any changes to the link after Fabric or Tenant formation, the RASlog service receives the syslog message.

The sequence of RASlog operations is as follows:

1. The RASlog Service processes the syslog message and notifies all services through message-bus.
2. The Inventory Service receives the RASlog Service message and updates relevant asset details in the database.
3. The Inventory Service notifies Fabric and Tenant Services of any changes in the configurations.
4. Fabric and Tenant Services review the state changes and display information about any pending configurations.

   You can choose to update Fabric or Tenants for the current state.
5. When a device is deleted from the Inventory Service, EFA is unregistered as a syslog recipient from the device. If unregistration of EFA fails, deletion still proceeds.
6. The RASlog Service listens to Device Registration and Device Deletion messages to ensure that messages from registered devices are not dropped.

Notification Service

Notifications are alerts, tasks, and events that EFA sends to subscribers by HTTPS webhook or syslog over RELP (Reliable Event Logging Protocol).

Overview

All notifications are derived from the syslog events that are received from the devices that EFA manages.

Alerts are notifications that EFA services send for unexpected conditions, such as the following:

- Loss of switch connectivity
- Failure to configure the fabric, tenant, or EPG on the device
- Failure to perform operations such as port up or port down, set speeds, and breakout mode
- Firmware download failure
- Devices exiting maintenance mode

Task notifications are based on user-driven or timer-based operations, such as the following:

- Registering or updating a device
- Device timer collection completed
- Adding devices to a fabric
- Creating, updating, or deleting a fabric
- Creating, updating, or deleting a tenant
- Creating, updating, or deleting an endpoint group (EPG)

Notification methods

EFA supports two methods of notification: HTTPS webhook and syslog (using RELP over TLS). The format of the notifications is the same for both methods. You can configure one or both methods.

Webhook

This REST API-based method is a POST operation. The notification payload is in the body of the HTTPS call. Use the **`efa notification subscribers add-https`** command to register a subscriber for this method of notification.

Syslog over RELP

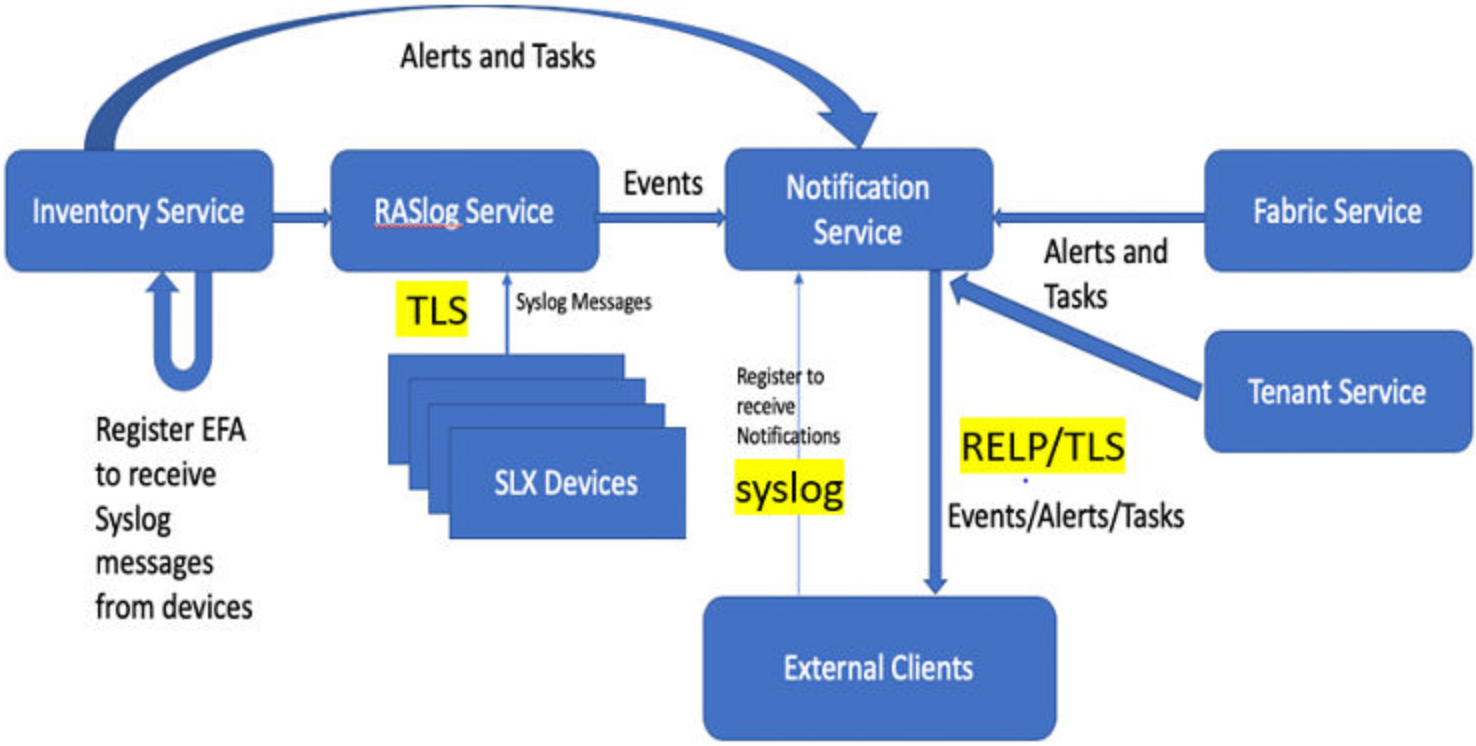
In this client-server method, the client initiates the connection and the server listens. In this scenario, the client is the Notification service and the server is the remote system where syslog is configured to work with RELP. Any external server that is configured with RELP can be registered as a subscriber to EFA notifications.

When RELP is configured with mTLS, EFA must be installed in secure mode. For more information, see the 'EFA Installation Modes" topic in the Extreme Fabric Automation Deployment Guide, 2.4.0.

Communication from SLX devices occurs over TLS. The certificates required for SLX devices to work with secure syslog are generated when the devices are registered.

Use the **`efa notification subscribers add-syslog-relp`** command to register a subscriber for this method of notification.

Notification workflow



In general, the order of operations is as follows:

1. For each SLX devices that is registered in EFA, EFA registers itself as a syslog receiver on each device.
2. The RASlog service receives syslog messages from the registered devices over TLS.
3. Messages are processed and converted to JSON, a format that is easier to search, sort, and perform operations on.
4. The Inventory service, Fabric service, and Tenant service send alerts and tasks to the Notification service.
5. External clients (subscribers) receive notifications by HTTPS webhook, syslog (using RELP over TLS), or both, depending on the subscription.

Notification example

In this example, the creation of a tenant sends a task notification to the syslog server.

Here, the tenant is created.

```
$ efa tenant create --name tnt1
Tenant created successfully.
--- Time Elapsed: 167.716003ms ---
```

And here are the corresponding syslog messages.

```
<5>Aug 20 11:25:28 10.24.95.154(10.24.95.154)
{"application":"ts","device_ip":"","message":"Tenant create request received
:request={\"name\":\"tnt1\"}","scope":"user","status":"started","task":
"EFA-001001","timestamp":"2020-08-20T13:42:55Z","type":"Task","username":"extreme"}
```

```
<5>Aug 20 11:25:28 10.24.95.154(10.24.95.154)
{"application":"ts","device_ip":"","message":"Tenant create request success
:request={\"name\":\"tnt1\"}","scope":"user","status":"succeeded","task":
"EFA-001002","timestamp":"2020-08-20T13:42:55Z","type":"Task","username":"extreme"}
```

Event Synchronization and Missing Event Handling

Provided VNI: 1006 already consumed in fabric

Symptom	Condition	Workaround
EPG create fails with VNI resource not being available in the fabric.	Execute EPG create, delete, and re-create CLI in quick succession: 1. Create EPG/Networks with user-provided VNI parameter. 2. Delete EPG. 3. Create EPG again with the same parameters as in step 1.	Provide a wait of 30 seconds between the create and delete CLI on the same EPG.

vrf delete from epg and re-adding vrf to epg fails intermittently

Symptom	Condition	Workaround
EPG update "vrf-add" operation fails with the reason as VRF to be added has conflicting VRF on the switch.	Execute EPG update "vrf-add", "vrf-delete", and "vrf-add" operation CLI in quick succession: 1. Update EPG for operation vrf-add. 2. Update EPG for operation vrf-delete. 3. Update the same EPG again with operation vrf-add for the same VRF which was deleted in step 2.	Wait of 30 seconds between the EPG update vrf-add and vrf-delete operations on the same EPG.

vrf_route_target_mapping error while creating epg (after vrf delete)

Symptom	Condition	Workaround
When VRF is added and deleted to/from and Endpoint Group, in quick succession, multiple times, events received from inventory service can get interleaved with the commands. This causes EFA command execution path to find database entries that are yet to be deleted due to previous command run.	Issue is observed when vrf-add and vrf-delete operation is executed multiple times on Endpoint Group in quick succession.	Wait for a few minutes before executing the vrf-add again on Endpoint Group.

Inventory device update fails to change device table router-ip field

Symptom	Condition	Workaround
If the router-id is changed on the device (add/delete/modified), the device table router-ip field will not change after you do inventory device update.	It's found that inventory did detect the diff and generate the RouterBgpUpdatedMsg to Tenant. Tenant did not handle this message.	Add hook in the RouterBgpUpdatedMsg on tenant handler to set the RouterIP, so that the Tenant DB will have the same value with the device after update.

REST opertations aren't retried (as applicable) during the service boot

Symptom	Condition	Workaround
REST opertations aren't retried (as applicable) during the service boot up.	The status aren't set for all the REST operations AFTER publishing all the necessary events on the message bus.	For all the REST operations, set the status AFTER publishing all the necessary events on the message bus.

EFA as SNMP Proxy

EFA acts as the SNMP Manager for all the SLX devices and agents and receives the traps from all the devices in its inventory.

Simple Network Management Protocol (SNMP) traps are alert messages sent from a remote SNMP-enabled device to a central collector, the SNMP Manager. Trap messages are the main form of communication between SNMP monitoring tools – an SNMP Agent and an SNMP Manager.

EFA acts as the SNMP Manager for all the SLX devices and agents and receives the traps from all the devices in its inventory. Once you register an SLX device with EFA, EFA automatically configures the SLX device to send v3 traps to EFA.

EFA acts as an SNMP proxy for all the SNMP v2 and v3 traps received from the SLX devices, forwarding them onto an external trap receiver, if there is one.

- EFA subscribes to be a v3 trap receiver with a predefined v3 user name, authentication key, and privacy key.
- If you set up EFA to be a v2c trap receiver, you must provide a community string.

During an update operation, EFA verifies that it is still registered to receive traps from the SLX devices. If a device is unregistered from EFA, the SNMP configuration on the device is updated to no longer send traps to the EFA IP address.

Notes

- The device IP address is the one included in `SNMP-COMMUNITY-MIB::snmpTrapAddress.0`. It is not the EFA IP address.
- EFA forwards all received traps. In other words, no trap is filtered out.
- Port 162 on the host where EFA is installed must be available. During a fresh installation, the port availability is checked and the installer returns an error if the port is not available. However, during an upgrade from a previous version of EFA, you must ensure that the port is free.

For more information about SLX-OS MIBs, see the *Extreme SLX-OS MIB Reference* for your version of SLX-OS.

Limitations

- A maximum of four trap subscribers is supported.
- V2c and v3 SNMP subscribers are not validated.
- Only traps generated by SLX devices are forwarded. Alerts and alarms from EFA itself are not forwarded.
- Only traps are forwarded. Current EFA tasks or alerts and syslog messages are not forwarded as traps.
- SNMP Informs are not supported.
- There is no in-band support for trap forwarding.
- The Drift and Reconcile process does not show a drift in device configuration for SNMP v3 trap configuration that EFA has pushed. However, every time the device update operation runs, EFA checks if the device is configured to send traps to EFA and if not, pushes the configuration again.
- For a multi-node deployment during failover of the active node, some traps might be missed while the SNMP service is bootstrapping on the new active node. There is no loss of traps if the standby node goes down.

gosnmp-service

The gosnmp-service is responsible for persisting the trap subscribers, receiving the SNMP traps, and forwarding them to the subscribers.

The service is stateless, so no historical data (that is, previously received traps) will be persisted.

For high availability deployment, the service will be running in active/active mode, however since the VIP is bound to one host at a time, the pod running on the active node will be the one receiving the traps. On failover, the standby node takes over and the SNMP service running on that node will forward the traps.

You may have multiple IP subnets configured to access EFA. In such a case, EFA creates multiple subinterfaces under the management interface to which EFA is bound. EFA does not determine which interface sends out the trap, syslog or webhook. The administrator is responsible for configuring a route to the recipient. If one is found, the server will send out the trap. For more information, see Multiple Management IP Network on page 46.